Høgskulen på Vestlandet

BACHELOROPPGAVE

Automatisert merking ved ansiktsgjenkjennelse

Automated Labelling by Facial Recognition

Markus Pedersen

Matias Raknes

Stian Trohaug

Dataingeniør og informasjonsteknologi

Fakultet for ingeniør- og naturvitskap

Jerry Chun-Wei Lin

Pål Ellingsen

22.05.2023

TITTELSIDE FOR HOVEDPROSJEKT

Rapportens tittel: Automatisert merking ved ansiktsgjenkjennelse	*Dato:* 22.05.2023
Forfatter(e): Markus Pedersen, Matias Raknes, Stian Trohaug	*Antall sider u/vedlegg:* 57 *Antall sider vedlegg:* 106
Studieretning: Dataingeniør og informasjonsteknologi	*Antall disketter/CD-er:* 0
Kontaktperson ved studieretning: Jerry Chun-Wei Lin, Pål Ellingsen	*Gradering:* Ingen
Merknader:	

Oppdragsgiver: Vizrt	*Oppdragsgivers referanse:* EB-13
Oppdragsgivers kontaktperson: Roger Sætereng	*Telefon:* 92215303

Sammendrag:

Denne rapporten tar for seg planleggingen, arbeidet og dokumentasjonen for utvikling av en programvare for ansiktsgjenkjenning av mennesker på direkte TV-sendinger. Dette prosjektet er et resultat av kunnskapen og erfaringene som gruppemedlemmene har opparbeidet gjennom deres bachelorgrad på HVL.

Stikkord:

Ansiktsgjenkjenning	Kringkasting	Maskinlæring

FORORD

Denne rapporten dokumenterer arbeidet rundt prosjektet som tar sikte på å utvikle en prototype kalt Automated Labelling by Facial Recognition (Automatisk merking ved ansiktsgjenkjenning). Hovedmålet med prosjektet var å utvikle et program som kan oppdage og gjenkjenne ansikter i en live-TV-sending og automatisk merke dem med relevant informasjon.

Det gis en stor takk til utviklerselskapet Vizrt for deres hjelp som oppdragsgiver, både med relevant erfaring og bakgrunn, samt støtte med kontorplass og utstyr. Det vil også uttrykkes stor takknemlighet til prosjektets veiledere, Jerry Chun-Wei Lin med doktorgradsstudenter for kompetanse innenfor maskinlæring og kunstig intelligens, og Pål Ellingsen, for gode råd og verdifull veiledning gjennom prosjektet.

Dette prosjektet har gitt gruppen en unik mulighet til å anvende sine teoretiske kunnskaper i praksis, og har bidratt til både faglig og personlig utvikling. Det håpes at resultatene av dette arbeidet vil være verdifullt for både akademisk og industriell bruk.

INNHOLDSFORTEGNELSE

1 INNLEDNING

Målet med dette kapitlet er å gi en introduksjon til opphavet for prosjektet. I de kommende underkapitlene vil kontekst, motivasjon, prosjekteier, problembeskrivelse og mål bli utdypet.

1.1 Kontekst

I denne delen vil det bli gitt en beskrivelse av den konteksten som ligger til grunn for gruppens oppgave. Gruppens oppdragsgiver - Vizrt - vil bli presentert, og en beskrivelse av deres behov for et automatisert system for oppdaging og gjenkjenning av personer på direktesendt TV.

En utfordring for Vizrt og deres kunder ligger i å oppnå presis og effektiv identifisering og gjenkjenning av personer i sanntid på direktesendt TV. Dette hjelper for å opprettholde en jevn og sømløs kringkasting. Dette er en utfordrende prosess, særlig når det gjelder situasjoner med flere personer i bildet samtidig. Som beskrevet av Roger R. Sætereng, gruppens interne veileder og R&D (Research and Development) manager for Vizrt, har denne prosessen tradisjonelt blitt utført manuelt av en eller flere personer som overvåker sendingen og legger inn relevante etiketter for hånd.

I denne rapporten vil det undersøkes teknikker for ansiktsgjenkjenning og evaluere deres effektivitet basert på gruppens problemstilling.

1.2 Motivasjon

Formålet med dette prosjektet er å utvikle og implementere en prototype for å oppdage, spore og gjenkjenne ansikter på direktesendt TV, samt evaluere dens effektivitet med hensyn til prosjektets problemstilling. Med økende tilgjengelighet av medier, vil automatisering av dette aspektet ved kringkastingsprosessen være en viktig faktor for å effektivt skape mer relevant innhold for seerne.

Ved å teste og evaluere prototypens effektivitet og brukervennlighet, vil gruppen kunne bidra til utviklingen av et nytt og innovativt verktøy for automatisering av visse aspekter i dagens kringkastingsindustri.

1.3 Prosjekteier

Som nevnt tidligere er Vizrt oppdragsgiveren for dette prosjektet, og deres behov for et automatisert system for oppdaging og gjenkjenning av personer på direktesendt TV danner grunnlaget for oppgaven vår.

Vizrt ble etablert i Bergen i 1997 og har siden utviklet seg til å bli en verdensledende leverandør av avanserte teknologiske løsninger til kringkastingsindustrien. Selskapet har en

bred produktportefølje som spenner fra virtuelle studioer til sanntidsgrafikk og automatisk produksjon. Bedriften har også et globalt kundegrunnlag som inkluderer store mediehus som CNN, BBC og NBC (Vizrt, u.å.).

Som en ledende aktør i bransjen, er det naturlig at Vizrt stadig ønsker å forbedre og utvikle sine produkter og tjenester. Etter hvert som kringkastingsindustrien fortsetter å utvikle seg, og nye teknologier og plattformer blir introdusert, er det viktig for Vizrt å være i forkant av utviklingen og tilby innovative løsninger til kundene.

1.4 Problembeskrivelse og mål

Problemstillingen for dette prosjektet er relatert til behovet for en effektiv og nøyaktig metode for å oppdage, gjenkjenne og spore personer på direktesendt TV. Som tidligere nevnt har denne prosessen tradisjonelt blitt utført manuelt av en eller flere personer som nøye overvåker kringkastingen. Imidlertid er det et potensial for forbedring gjennom automatisering, som kan bidra til å effektivisere og forenkle denne oppgaven.

En utfordring med automatisering av de nevnte funksjonalitetene på direktesendt TV er tilstedeværelsen av flere personer samtidig i bildet, noe som gjør prosessen mer ressurskrevende. Videre kan skiftende synsvinkler fra kamera til kamera føre til at personer endrer posisjon i bildet, som også utgjør en betydelig utfordring.

Målet med dette prosjektet er å utvikle en prototype som kan oppdage, spore og gjenkjenne ansikter i direktesendt TV, og deretter evaluere prototypens effektivitet i forhold til problemstillingen. Gjennom undersøkelser av ulike teknikker for ansiktsoppdaging og gjenkjenning, samt evaluering av prototypens effektivitet og brukervennlighet, vil det legges til rette for at oppdragsgiveren kan utvikle et innovativt verktøy som kan ha en positiv innvirkning på kringkastingsindustrien.

1.5 Oppbygging av rapporten

I innledningen (kapittel 1) er det en introduksjon og gjennomgang av rapporten. Dette kapitlet beskriver prosjektets bakgrunn, motivasjon og mål. Her settes det også lys på problemstillingen for oppgaven. Kapittel 2 inneholder praktisk bakgrunn, avgrensninger, hvilket ressurser det er behov for og litteratur om problemstillingen for oppgaven. Kapittel 3 beskriver designet av prosjektet, ved å drøfte forslag til løsninger for oppgaven, og diskutere valgt løsning. Kapittel 3 tar også for seg ulike verktøy gruppen ser for seg å bruke, det drøftes også potensielle utviklingsmetodikker. I kapittel 4 blir metodene for å løse prosjektet beskrevet. Det gis en detaljert oversikt over løsningen og arbeidet som ligger til grunn for prosjektets resultat. Videre følger kapittel 5, som viser til resultatene av den valgte løsningen i kapittel 4. Prosjektets resultater drøftes med hensyn til evaluering og funksjonelle krav. Kapittel 6 diskuterer de ulike valgene gruppen har tatt og drøfter utfallet av disse. I kapittel 7 presenteres det en konklusjon for prosjektet og forslag til videre arbeid. Kapittel 8 inneholder referanselisten. Kapittel 9 viser til vedlegg og kapittel 10 inneholder en ordliste som beskriver ulike begrep og ord brukt i rapporten.

2 PROSJEKTBESKRIVELSE

Dette kapitlet tar for seg en detaljert beskrivelse av prosjektet og dets bakgrunn.

2.1 Praktisk bakgrunn

Den praktiske bakgrunnen for prosjektet omhandler et automatisert system for oppdaging og gjenkjenning av personer på direktesendt TV. Et ideelt automatisert system vil kunne bidra til å redusere risikoen for feil og samtidig frigjøre ressurser til andre oppgaver.

2.1.1 Tidligere arbeid

Tidligere arbeid inkluderer oppdragsgivers utvikling av en prototype som implementerer ansiktsgjenkjenning og -sporing på videofiler ved bruk av programmeringsspråket Python og maskinlæringsmodellen MTCNN. Selv om denne prototypen har blitt brukt som referanse for videre utvikling, har den blitt forkastet som en fullstendig løsning grunnet utfordringer angående hastighet og effektivitet ved sammenligningen av ansiktsrepresentasjoner (embeddings). Derimot vil prosjektet benytte seg av denne prototypen som et hjelpemiddel for å utvikle en egen implementasjon, som tar høyde for hvert av kravene i prosjektet.

Det er verdt å merke seg at MTCNN-algoritmen som ble brukt i den nevnte prototypen, ikke er en original utvikling, men en implementasjon av en eksisterende teknologi. MTCNN-algoritmen er implementert i facenet_pytorch, som er en implementasjon av en rekke dyp læringsmodeller for ansiktsgjenkjenning (Esler, 2023, avsnitt 1) hvor kildekoden er fritt tilgjengelig. Teknologiske fremskritt innenfor dyplæring for ansiktsgjenkjenning har gjort det mulig å utvikle sofistikerte og nøyaktige algoritmer som kan brukes til mange formål. I lys av dette har oppdragsgiveren allerede benyttet eksisterende teknologi for å utvikle en prototype som kan brukes som grunnlag for videreutvikling av et mer avansert programvaresystem.

2.1.2 Initielle krav

Det skal utvikles en prototype for et programvaresystem som har til hensikt å automatisere visningen av "lower third graphics" under direktesendt TV. Denne grafikken har som hensikt å vise navnene på personene som befinner seg i bildet. Programmet skal inkorporere eksisterende modeller for ansiktsgjenkjenning og sporing. Dette avsnittet presenterer kravene som stilles til programvaren.

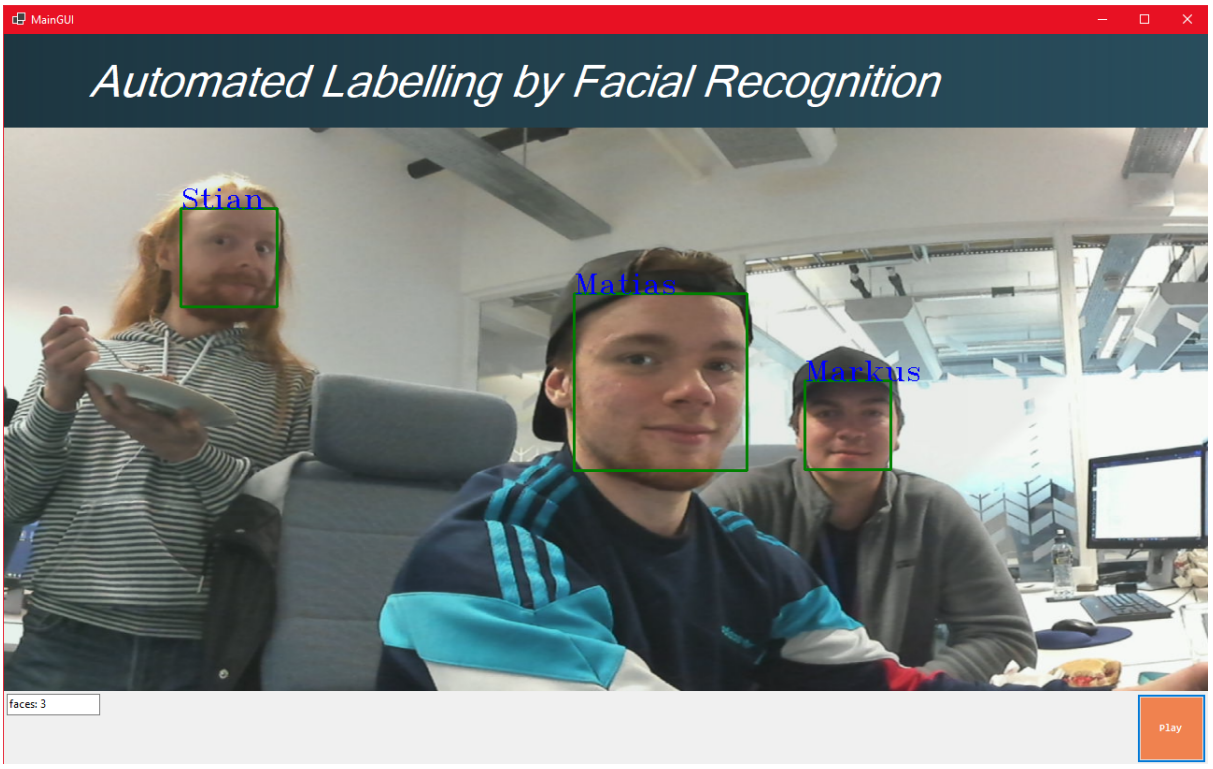
Programmet skal:

- Utføre automatisert ansiktsgjenkjenning og merking i sanntid for å identifisere personer på et live kamera.
- Kunne presentere nøyaktig informasjon på skjermen.
- Være lett å styre fra et brukergrensesnitt.

2.1.3 Initiell løsnings-idé

Figur 2.1 viser en tidlig iterasjon for funksjonaliteten til programmet.
Programmet mottar 30 bilder i sekundet fra et kamera og forbereder dataene slik at de kan prosesseres av modellen for ansiktsgjenkjenning. Dersom modellen detekterer ett eller flere ansikter, vil det genereres en embedding for hvert ansikt som representerer det gitte ansiktet. Programmet oppretter da "bounding-boxes", som avgrenser området ansiktet befinner seg innenfor. Posisjonen til disse boksene brukes for sporing, slik at modellen ikke skal behøve å sammenligne det samme ansiktet med seg selv dersom det ikke har flyttet seg så mye at det er grunn til mistanke for at det er et nytt ansikt. Denne sporingen av forflytning er noe som bidrar til økt effektivitet da produksjon og sammenligning av embeddings krever en del ressurser i form av prosessorkraft.

Programmets brukergrensesnitt er til for å kunne forandre på etiketter, overvåke og eventuelt rette opp feil. Hovedfokuset vil være på programmet som kjører modellen, da eventuelle kunder kan ha egne systemer for brukergrensesnitt. Det er forventet at programmet kjøres på systemer med tilstrekkelig maskinvare, slik at CPU og GPU kan benyttes for å oppnå ønsket resultat. Programmet vil lytte kontinuerlig til bildestrømmen fra kameraet og vedlikeholde og oppdatere informasjonen som vises på skjermen.



Figur 2.1: Tidlig iterasjon av brukergrensesnitt

2.2 Avgrensninger

I dette prosjektet er det definert avgrensninger på flere områder:

- Det er valgt å bruke tidligere modeller for ansiktsgjenkjenning i stedet for å lage egne modeller.
- Det er valgt å kun bruke ansiktsgjenkjenning for å sette på etiketter på personer, ikke for automatisering av styring av kameraer eller lignende.
- Det er valgt å fokusere på sammenhengende videoopptak uten brå endringer i kameraperspektiver - såkalte "jump-cuts."

Avgrensningene vil bidra til å holde prosjektet fokusert og realistisk og vil også bidra til å redusere omfanget av oppgaven for å gjøre det mer gjennomførbart innenfor den gitte tidsrammen.

2.3 Ressurser

For å kunne utføre presis ansiktsgjenkjenning er en datamaskin med tilstrekkelig prosessorkraft og minne avgjørende. Gruppen har behov for en datamaskin med en kraftig GPU.

Av gruppens erfaring er det nødvendig å ha tilgang til ulike programvarebibliotek for å skrive en programvare som dette. Ansiktsgjenkjenning er en komplisert prosess med mange parametre, og det ville krevd mye tid å bygge et slikt program fra bunnen av. Ved å bruke et programvarebibliotek kan gruppen dra nytte av allerede eksisterende kode som er testet og pålitelig.

For å utvikle en programvare må man ha tilgang til et integrert utviklingsmiljø (IDE) for å skrive kode.

Gruppen har fått tildelt kontorplass hos oppdragsgiveren. Dette gjør det enkelt å søke råd, veiledning og kompetanse fra erfarne fagfolk som har vært i bransjen i mange år. Andre personer gruppen anser som gode ressurser for oppgaven er gruppens veiledere, Jerry Chun-Wei Lin (med studenter) og Pål Ellingsen.

2.4 Litteratur om problemstillingen

2.4.1 Kunstig intelligens

Kunstig intelligens (KI) er et felt innen datavitenskap som handler om simulering av intelligens i maskiner. Designet til disse maskinene har som hensikt å etterligne menneskelig atferd og læring.

En undergren av KI er maskinlæring (ML). ML-systemer forutsier fremtidig utfall eller beslutninger ved å benytte lærdom fra tidligere erfaringer (IBM, u.å.). Innen ML finnes det

igjen flere underkategorier, men for dette prosjektet er det tilstrekkelig å nevne overvåket læring. Overvåket læring benytter data med kjent innhold. Modellen bruker så dette for å forutsi utfall på ukjent data. I konteksten av dette prosjektet, betyr dette at modellen lærer av navngitte personer.

Dyp læring er igjen en underkategori av maskinlæring som bruker nevrale nettverk med mange lag, eller "dype" nettverk, for å lære fra store datamengder. Dype nettverk er spesielt effektive for oppgaver som bildegjenkjenning (Boesch, 2023, avsnitt 1).

For å fremme forståelsen av dette prosjektet, vil noen sentrale begreper bli definert: klassifisering, regularisering, data augmentering, forhåndsbehandling og trenings epoker (Google, 2023).

- Klassifisering: En sentral oppgave innen overvåket læring med mål om å identifisere kategorien, eller klassen, til en ny, ukjent instans basert på læring fra tidligere data.
- Regularisering: Denne teknikken brukes i maskinlæring for å forhindre overtilpasning, som oppstår hvis modellen blir for kompleks og presterer dårlig på ukjent data. Ved å bruke regularisering kan modellens kompleksitet styres, som bidrar til å balansere modellens evne til å lære fra treningsdata og generalisere til nye data.
- Data Augmenting: Denne teknikken innebærer å øke mengden treningsdata på kunstig vis, gjennom modifikasjoner av tilgjengelig data. Disse modifikasjonene kan være rotasjon, skalering, beskjæring og tillegg av støy. Dette kan forbedre modellens robusthet og evne til å generalisere til nye data.
- Forhåndsbehandling: Dette er de nødvendig trinnene som tas for å forberede rådata for trening av modeller. Hva disse trinnene innebærer vil avhenge av den spesifikke oppgaven. Noen eksempler er fjerning av støy, normalisering og håndtering av manglende data.
- Trenings Epoker: En epoke innen maskinlæring er en fullstendig gjennomgang av treningsdata en gang under trening. Antall epoker er det samme som antall ganger modellen vil arbeide gjennom hele datasettet. Ved å øke antall epoker kan modellens ytelse øke, men for mange epoker kan føre til overtilpasning.

Gjennom en grunnleggende forståelse av disse begrepene, vil det være lettere å sette anvendelsen av kunstig intelligens i dette prosjektet i kontekst

2.4.2 Utvikling og anvendelse av kunstig intelligens

KI har de siste årene hatt en stor økning i popularitet. I 2020 viste en undersøkelse publisert av Eurostat at syv prosent av alle norske foretak med minst ti sysselsatte brukte kunstig intelligens (Eurostat, 2021). En tilsvarende undersøkelse gjort av SSB, viser en økning fra syv til elleve prosent på ett år (Walther-Zhang, 2021, avsnitt 1).

I en artikkel publisert av Abby McCain kommer det frem at kunstig intelligens har en positiv innvirkning på bedrifters omsetning, og 91,5 prosent av ledende foretak investerer i KI (McCain, 2022, avsnitt 1). Disse tallene viser hvordan utviklingen av kunstig intelligens har økt de siste årene, og at det antageligvis vil fortsette å øke da investeringene er så omfattende.

KI-teknologi brukes i en rekke bransjer, bl.a. helse, transport, finans og detaljhandel (Q.ai, 2023). KI-teknologi har gjort det mulig å forbedre eksisterende systemer og å utvikle nye og innovative løsninger som har vært utfordrende eller umulige å løse tidligere.

Mange bedrifter bruker nå KI-verktøy til å automatisere oppgaver, forbedre effektiviteten og skape en mer skalerbar virksomhet. KI har også blitt integrert i mange forretningsprosesser for å optimalisere kundeservice og forbedre kundeopplevelsen (Chaturvedi og Verma, 2022, avsnitt 3).

2.4.3 Kunstig intelligens innen kringkasting

Kringkastingsindustiren er en industri i stadig utvikling for å tilfredsstille seernes behov, og kunstig intelligens brukes allerede til flere formål som, forbedring av lyd og bilde, finne aktuelle nyhetssaker og talegjenkjenning (Gentile, 2020, avsnitt 5). Prosjektets oppdragsgiver ser derfor et behov og en mulighet til å implementere dette for å automatisere visse deler av en direktesending. Oppdragsgiveren for prosjektet - Vizrt - har publisert en artikkel “5 ways Artifical Intelligence and Machine Learning will make sports broadcasting smarter” (Torsvik, u.å.), hvor de forteller at kunstig intelligens og maskinlæring kan bidra til at medieselskaper kan oppnå fordeler som tidligere har vært ansett som uoppnåelig.

3 DESIGN AV PROSJEKTET

Dette kapitlet handler om ulike aspekter ved design og planlegging av prosjektet, inkludert valg av løsning, verktøy og prosjektmetodikk. Kapitlet gir en oversikt over ulike alternativer og gir begrunnelser for valgene som har blitt tatt. Det gis også en risikovurdering og en evalueringsplan for prosjektet. Kapitlet skal gi en grundig forståelse av prosjektets utforming og ressursbruk som kreves for å fullføre det.

3.1 Forslag til løsning

Dette prosjektet krever et utvalg av teknologier og metoder som kan gi den mest effektive løsningen for ansiktsgjenkjenning i sanntid. Denne seksjonen vil beskrive de vurderte alternativene, som applikasjonstype og maskinlæringsmodeller, samt gi en oversikt over fordelene og begrensningene hver av dem tilbyr.

3.1.1 Vurdering av applikasjon

Applikasjonen må enten være en web-basert applikasjon eller en desktop-applikasjon. Gruppens valg av enten web-basert eller desktop-basert applikasjon vil være basert på applikasjonens krav og effektivitet, da hele applikasjonen vil være forankret i én av disse to alternativene.

Web-basert applikasjon

En web-basert applikasjon er en applikasjon som kjører i en nettleser, og som er tilgjengelig fra en hvilken som helst enhet som har tilgang til internett.

- Tilgjengelighet: Ettersom en web-basert applikasjon er tilgjengelig fra en hvilken som helst enhet med internett-tilkobling er det ikke nødvendig å installere applikasjonen på hver enkelt datamaskin. Dette gjør det enklere for brukerne å få tilgang til applikasjonen (Rana, 2021, avsnitt 2).

- Oppdateringer og vedlikehold: En web-basert applikasjon kan oppdateres sentralt og automatisk (Rana, 2021, avsnitt 2). Det er ikke nødvendig å distribuere oppdateringer til hver enkelt datamaskin som har installert applikasjonen. Dette gjør at en web-basert applikasjon kan være mer oppdatert og sikker enn en desktop-applikasjon.

- Skalerbarhet: Det er lettere å skalere en web-basert applikasjon etter hvert som antallet brukere øker. En desktop-applikasjon kan være begrenset av maskinvaren til datamaskinen den kjører på, mens en web-basert applikasjon kan kjøres på en server og skaleres etter behov.

- Kostnadseffektivitet: Det kan være mer kostnadseffektivt å utvikle en web-basert applikasjon enn en desktop-applikasjon. Det er ikke nødvendig å utvikle applikasjonen for flere plattformer og operativsystemer, og utviklingskostnadene kan dermed reduseres (Rana, 2021, avsnitt 2).

Desktop-applikasjon

En desktop-applikasjon er en applikasjon som må installeres på hver enkelt datamaskin for å kunne brukes.

- Bedre ytelse: En desktop-applikasjon kan ha bedre ytelse og raskere responstid enn en web-basert applikasjon fordi den kjører direkte på maskinvaren til datamaskinen og ikke er avhengig av nettleser eller internettforbindelse (Rana, 2021, avsnitt 2).
- Bedre tilpasningsmuligheter: Desktop-applikasjoner kan være mer tilpasningsdyktige og gi en bedre brukeropplevelse enn web-baserte applikasjoner. Dette kan være spesielt viktig for komplekse applikasjoner som krever mye interaksjon og tilpasning.
- Enklere tilgang til systemressurser: Desktop-applikasjoner kan enklere få tilgang til systemressurser som harddisk, minne og prosessor, som kan være nødvendig for å utføre visse oppgaver eller funksjoner (Rana, 2021, avsnitt 2).
- Bedre sikkerhet: Desktop-applikasjoner kan være mer sikre enn webapplikasjoner fordi de ikke er avhengig av nettleseren og nettleserens sikkerhetssystemer. Dette kan redusere risikoen for nettangrep og datatyveri (Rana, 2021, avsnitt 2).

3.1.2 Vurdering av modeller

Ansiktsgjenkjenning i sanntid er en krevende prosess, og gruppen har vurdert flere ulike modeller for å finne den mest effektive løsningen. Gjennom utviklingen av prototypen har det blitt implementert og vurdert ulike modeller.

MobileNetV2

- Dette er en lettvekts modell som er spesielt designet for å kunne yte godt selv med begrensede ressurser. MobileNetV2 sin høye hastighet og lave ressursbruk, gjør den til en god løsning for applikasjoner i sanntid (Sandler *et al.*, 2018).

LBPH (Local Binary Patterns Histogram)

- Dette er en enkel og effektiv metode for ansiktsgjenkjenning. Den er mindre kompleks og raskere enn dyplæringsmetoder, men den er igjen mindre nøyaktig og mer følsom for variasjoner i ansiktsuttrykk og belysning (Salton do Prado, 2017, avsnitt 1).

ResNetV2

- Denne modellen er kjent for sin høye ytelse innen bildegjenkjenning. Den lager representasjoner av ansikter (embeddings) i sanntid, noe som potensielt kan forbedre effektiviteten. Men modellens kompleksitet kan føre til problemer relatert til ytelse (Dulčić, 2020, avsnitt 3).

3.2 Valgt løsning

Når man utvikler en applikasjon, må man ta en rekke avgjørelser som kan påvirke hvordan applikasjonen fungerer og hvilke fordeler den har for brukerne. En sentral beslutning som må tas tidlig i prosessen, gjelder valget mellom en nettbasert applikasjon og en skrivebordsapplikasjon. Hver av disse løsningene bærer med seg distinkte fordeler og ulemper som må vurderes nøye. I tillegg til dette, for et prosjekt som innebærer sanntids ansiktsgjenkjenning, er valget av en passende modell viktig.

3.2.1 Valg av applikasjonstype

Etter grundig evaluering av de nevnte strategiene, har gruppen besluttet å utvikle en desktop-applikasjon. Dette valget er motivert av en rekke fordeler som gruppen anser som viktig for denne typen prosjekter.

Hovedelementene i ansiktsgjenkjenning, nemlig behandling av store datasett og implementering av avanserte algoritmer, kan potensielt medføre økte lastetider og dårlig responsivitet i en web-applikasjon. Disse utfordringene kan lettere omgås med en desktop-applikasjon, da denne vil kjøre direkte på datamaskinens maskinvare.

I tillegg gir desktop-applikasjoner bedre tilgang til systemressurser, noe som kan være avgjørende for optimalisering av visse funksjoner innen ansiktsgjenkjenning (OliaG et.al., 2022, avsnitt 7). Fra et sikkerhetsperspektiv kan desktop-applikasjoner tilby bedre beskyttelse, ettersom de ikke er underlagt nettleserens sikkerhetssystemer (Ortiz, 2022, avsnitt 5).

Gruppens tidligere erfaring med utvikling av desktop-applikasjoner har også veid inn i beslutningen. Gruppemedlemmene er trygge på at deres eksisterende kunnskapsbase og ferdigheter vil muliggjøre utviklingen av en pålitelig og effektiv desktop-applikasjon.

3.2.2 Valg av modell

Gjennom utviklingen av prosjektet var valget av modell en viktig avgjørelse. Gruppen tok tidlig en avgjørelse om å utvikle prototypen i C#, med et ønske om å forbedre ytelsen sammenlignet med prototypen presentert av oppdragsgiveren. Dette medførte visse begrensninger for valg av modell, da det gjør implementasjonen litt mer komplisert.

MobileNetV2 skiller seg ut med sin relativt lave ressursbruk og høye hastighet. Denne modellen er, som nevnt, designet for å kunne yte godt med begrensede ressurser. LBPH er en enklere metode, men tilbyr en lavere nøyaktighet og er mer følsom for variasjoner. ResNetV2 er kjent for sin høye ytelse, men dens kompleksitet kan føre til problemer med ytelse.

For å best kunne evaluere disse modellene, og ta en god beslutning, har alle de tre ovennevnte modellene blitt implementert og testet i applikasjonen. Etter dette besluttet gruppen å vektlegge implementasjonen av MobileNetV2. Dette valget var motivert av flere faktorer, men den lave belastningen på maskinvaren og nøyaktigheten til modellen var avgjørende. For

å forsikre at dette var en god besluttning, så ble alle tre modellene trent på det samme datasettet av gruppemedlemmene, og resultatene viste at MobileNetV2 presterte best.

3.3 Valg av verktøy

I dette prosjektet har en omfattende vurdering blitt utført med henblikk på verktøy som er nødvendige for å utvikle en operativ programvare som er i stand til å utføre identifisering av mennesker i en live broadcast gjennom bruk av ansiktsgjenkjenning, og etterfølgende påsetting av etiketter. Gruppen har valgt flere verktøy som vil kunne bidra til å realisere dette formålet.

3.3.1 Språk, rammeverk og IDE

For å utvikle programvaren for prosjektet, har gruppen valgt Visual Studio som IDE for C#-koden. Dette valget er basert på kjennskap til Visual Studio og at denne gir en integrert opplevelse for utviklingen med funksjoner for testing og feilsøking av koden på en enkel og effektiv måte (anandmeg *et.al.*, 2023, avsnitt 1). I tillegg vil Python benyttes under utviklingsprosessen, og for å koble sammen kode fra de to språkene vil gruppen benytte seg av Python.net. Dette verktøyet gir mulighet til å kjøre funksjoner skrevet i Python fra C#-koden.

For å kunne effektivt behandle bildedataene og bruke maskinlæringsmodellene, har gruppen valgt å benytte seg av OpenCV-biblioteket i både C# og Python. Dette biblioteket gir muligheten til å behandle bilder og videoer på en omfattende måte i begge programmeringsspråkene (OpenCV, u.å.). Gitt OpenCVs funksjonaliteter for behandling av direktesendt video, er det et velegnet verktøy for å oppfylle prosjektets formål.

3.3.2 Kommunikasjon og samarbeidsverktøy

I prosjektet har det blitt valgt å anvende Git og GitLab som versjonskontrollsystemer for å legge til rette for effektivt samarbeid og deling av kode innad i gruppen. Dette valget er begrunnet i systemets evne til å gi oversikt over endringer i koden samt å muliggjøre samarbeid på tvers av utviklingsoppgaver.

For intern kommunikasjon og samarbeid innad i prosjektgruppen vil det bli brukt Teams og Discord som kommunikasjonsverktøy. Disse verktøyene gir muligheten for enkel deling av filer og skjermer, samt chat-funksjonalitet. I tillegg vil Google Docs og Google Spreadsheets bli tatt i bruk for å dele dokumentasjon og holde oversikt over arbeidsoppgaver på en effektiv måte.

3.3.3 Andre verktøy

Som et kunstig intelligens verktøy kan ChatGPT være nyttig for å gi forslag til kodeløsninger og feilsøking (Levine, 2023, avsnitt 1). Det er viktig å merke seg at verktøyet ikke er perfekt og kan gi feilaktige forslag, så det er viktig å bruke det med forsiktighet og alltid

dobbeltsjekke forslagene før de implementeres. Gruppen anser ChatGPT som en verdifull ressurs, men vil bruke det med forsiktighet og forbehold.

Når det gjelder maskinvare, vil prosjektet ta i bruk en stasjonær datamaskin med en kraftig grafikkprosessor (GPU) tilgjengelig på Vizrt sitt kontor. En slik maskinvare er avgjørende for håndtering av intensiv og avansert prosessering, som er nødvendig for programmets formål. Dette skyldes at ansiktsgjenkjenning innebærer behandling av store mengder bildeinformasjon, og prosessering av denne datamengden krever en betydelig mengde prosesseringskraft. Derfor har valget av en kraftig GPU på en stasjonær datamaskin blitt ansett som hensiktsmessig for å møte prosjektets behov for maskinvare. Dette ble også nevnt tidlig i prosjektfasen av oppdragsgiver kunne være nødvendig for å utvikle et program som tar i bruk ansiktsgjenkjenning.

3.4 Prosjektmetodikk

Prosjektmetodikk er avgjørende for hvordan man skal styre og organisere et prosjekt på en strukturert måte. Dette delkapitlet tar for seg gruppens tilnærming til struktur av planlegging og utvikling.

3.4.1 Utviklingsmetodikk

Gruppen betraktet valget av utviklingsmetodikk som avgjørende og mente at det måtte tilpasses prosjektets særegne behov og formål på best mulig måte. Derfor diskuterte gruppen flere løsninger på utviklingsmetodikk i starten av prosjektet.

Drøfting av utviklingsmetodikker

Det finnes flere anerkjente utviklingsmetodikker, dette er for å tilpasse utviklingsprosessen til ulike prosjekter og situasjoner. Noen prosjekter er komplekse og krevende, og krever omfattende planlegging med god struktur. Utviklingsmetodikkene som ble drøftet innad i gruppen i oppstartsfasen beskrives under.

- Scrum
  Scrum er en smidig metodikk som er kjent for å være fleksibel og tilpasningsdyktig med hensyn på endringer og uforutsette hendelser som kan oppstå i løpet av utviklingsprosessen (Sokolova, 2021, avsnitt 8). I scrum, utføres planlegging ved begynnelsen av hver sprint. Scrum har faste tidsrammer som teammedlemmene må følge og arbeidet blir fordelt på en planlagt måte.

- Kanban
  Dette er en annen type agil metode som fokuserer mer på visuell styring av arbeidsprosessen (Wikipedia, 2023). Hver oppgave er representert i en fysisk eller virtuell tabell. Oppgavene flyttes fra en kolonne til en annen under prosessen etter hvert som de blir fullført. I motsetning til scrum, er det ingen faste tidspunkter for planlegging, og man kan legge til oppgaver i arbeidsflyten når det er nødvendig. I tillegg, er det ingen fordeling av oppgaver på en planlagt måte som i Scrum.

- Vannfallsmodellen
  Vannfallsmodellen er en sekvensiell modell som består av ulike faser (Lutkevich, 2022, avsnitt 1). Fasene innebærer planlegging, design, implementering, testing og vedlikehold. Hver fase må fullføres før man går over til neste. Vannfallsmodellen gir tydelige krav for hver fase, noe som bidrar til at teamet har en klar forståelse for hva som skal gjøres. I motsetning til Kanban og Scrum, gir vannfallsmodellen lite fleksibilitet i arbeidsprosessen.

- Informal Project Management
  Dette er en utviklingsmetodikk som ikke bruker strenge standarder eller regler for å styre et prosjekt. Denne metoden har fokus på åpen kommunikasjon og stor fleksibilitet. Det følger heller ingen omfattende dokumentasjon, og det er heller ingen klare roller eller ansvarsområder (Copper Team, 2018). Informal Project Management egner seg godt til mindre prosjekt som ikke krever omfattende planlegging eller dokumentasjon.

Gruppen har opparbeidet seg god erfaring med Scrum og anser det som en anerkjent metode blant utviklere. Ved vurderingen av potensielle risikoer knyttet til gjennomføringen av prosjektet, som beskrevet i kapittel 3.4.3, kom gruppen frem til at Scrum var et godt alternativ på grunn av sin tilpasningsdyktighet og rykte for å håndtere uforutsette hendelser effektivt. Imidlertid innebærer Scrum også faste tidsrammer og nøye planlegging av arbeidsoppgaver, noe som gruppen ikke prioriterte å investere tid i. Gruppen vil heller ha større frihet og kreativitet på grunn av prosjektets natur og kompleksitet.

Kanban har derimot ingen faste tidsrammer eller planlegging av arbeidsoppgaver for hver enkelt person. Dette kan gi gruppen større fleksibilitet til å reagere på endringer og tilpasse seg endrede krav. Det vil også bli brukt mindre tid på administrativt arbeid som å oppdatere og endre planer. Gruppen vurderte tid verdifullt for å utføre kvalitetsarbeid.

Vannfallsmodellen i motsetning til de andre metodikkene gir lite rom for fleksibilitet. Og endringer i prosjektforløpet kan gjøre at gruppen setter seg fast. Dette kan være et resultat av at man oppdager feil eller mangler i en senere fase av utviklingsprosessen.

Gruppen ville ha stor frihet og kreativitet i løpet av utviklingsprosessen, og med dette virket også Informal Project Management som en egnet metodikk for et prosjekt som dette.

Valg av utviklingsmetodikk

Både Scrum og Kanban ble vurdert som gode valg for utviklingsmetodikk.

Gruppen valgte imidlertid å benytte seg av den uformelle utviklingsmetodikken, Informal Project Management, som ble ansett som bedre tilpasset gruppens spesifikke oppgave og ressurser. Informal Project Management vektlegger fleksibilitet, kontinuerlig kommunikasjon og samarbeid mellom gruppemedlemmene (Copper Team, 2018). Gruppens beslutning ble også påvirket av tidligere erfaringer med å jobbe sammen på tidligere prosjekter, som spilte en viktig rolle i valget av utviklingsmetodikk. For at Informal Project Management skal

kunne fungere effektivt i praksis, er det en forutsetning at alle gruppemedlemmer har tillit til at alle vil bidra til prosjektet.

Å bruke en uformell utviklingsmetodikk kan være et alternativ til Scrum for mindre utviklingsteam. Med en mindre gruppe på kun 3 personer kan det være mer tids- og ressursbesparende å bruke en uformell tilnærming, hvor man prioriterer utviklingen av produktet på en fleksibel og smidig måte, fremfor å fokusere på formaliteter og prosedyrer. Det er imidlertid viktig å være oppmerksom på at en uformell tilnærming kan føre til en mindre strukturert utviklingsprosess, og man må være forsiktig med å miste oversikt over progresjonen og målene til prosjektet. Derfor må kommunikasjon og tett samarbeid være på plass.

For å støtte denne metodikken, har gruppen brukt en rekke verktøy for å håndtere kommunikasjon og samarbeid. Kombinasjonen av verktøyene beskrevet i kapittel 3.3 og den fleksible metodikken har bidratt til å skape et miljø der gruppemedlemmene kan arbeide effektivt sammen og nå de overordnede målene for prosjektet.

3.4.2 Prosjektplan

Under ligger det en tabell for prosjektplanen til gruppen.

	Aktivitet	Start	Slutt	Timer	Ansvarlig
1	Obligatoriske aktiviteter	13.01.23	08.06.23	253	
4	OA-4: Oppstartsmøte	13.01.23	13.01.23	2	Markus
5	OA-5: Møte med veileder	19.01.23	19.01.23	1	Markus
6	OA-6: Prosjekttittel, prosjekthåndbok itr.1	17.01.23	29.01.23	16	Stian
7	OA-7: Statusmøte med veileder, oppfølging av OA-6	30.01.23	05.02.23	1	Matias
8	OA-8-1: Analyse av tidligere bacheloroppgaver	01.02.23	08.02.23	12	Matias
9	OA-8-2: Midtveisrapport itr.1, prosjekthåndbok itr.2	30.01.23	26.02.23	24	Stian
10	OA-9: Statusmøte med veileder, oppfølging av OA-8-2	27.02.23	05.03.23	1	Markus
11	OA-10: Midtveisrapport	27.02.23	09.03.23	48	Alle
12	OA-11: Midtveispresentasjon	10.03.23	14.03.23	2	Alle
13	OA-12: Prosjektstatus, prosjekthåndbok itr.3	30.02.23	23.04.23	32	Alle
14	OA-13-1: Rapport utkast 1	01.03.23	24.04.23	12	Alle
15	OA-13-2: Rapport utkast 2	25.04.23	07.05.23	16	Alle
16	OA-14: Endelig rapport	08.05.23	22.05.23	80	Alle
17	OA-15: Refleksjonsnotat	20.05.23	31.05.23	2	Alle

18	OA-16: EXPO poster	20.05.23	02.06.23	1	Alle
19	OA-17: Presentasjon	08.06.23	08.06.23	3	Alle
2	Utviklingsprosjekt	13.01.23	22.05.23	318	
20	Bestemme oppdrag	13.01.23	17.01.23	2	Alle
21	Undersøke Vue rammeverk	18.01.23	*Ikke satt*	16	Markus
22	Undersøke Prototypen (Kildekode)	24.01.23	*07.02.23*	16	Matias, Markus
23	Programmering back end	08.02.23	05.05.23	88	Alle
24	Programmering front end	31.01.23	28.02.23	96	Matias, Stian
25	Reprogrammere prototypen(Ett ansikt om gangen)	08.02.23	15.02.23	38	Alle
26	Sammenkoble backenden med etiketter	16.02.23	24.02.23	16	Alle
27	Funksjonalitet i front end (Innhold i etikettene)	28.02.23	07.03.23	22	Matias
28	Integrere videoavspilling i front end	07.03.23	14.03.23	8	Stian
29	Reprogrammere back end til å kjøre i real-time	14.03.23	04.04.23	16	Markus, Stian
3	Dokumentasjon	13.01.23	22.05.23	242	
30	Prosjekthåndbok	13.01.23	22.05.23	16	Alle
31	Visjonsdokument	18.01.23	22.05.23	16	Alle
32	Kravsdokument	31.01.23	22.05.23	12	Alle
33	Systemdokumentasjon	01.03.23	22.05.23	14	Alle
34	Bachelorrapport	01.03.23	22.05.23	96	Alle
35	Midtveisrapport	30.01.23	09.03.23	88	Alle

Tabell 3.1: Prosjektplan

3.4.3 Risikovurdering

En systematisk tilnærming til risikovurdering har blitt benyttet for å identifisere potensielle problemer som kan oppstå i løpet av prosjektet. Ulike teknikker har blitt brukt for å identifisere risikoer, inkludert brainstorming og risikomatriser. I tillegg har gruppen tatt hensyn til tidligere erfaringer for å identifisere områder med høyere sannsynlighet for problemer. For en detaljert beskrivelse av risikoanalysen, se vedlegg 2 (Prosjekthåndbok).

Etter at risikoene er identifisert, vurderes sannsynligheten for at hvert enkelt problem vil oppstå og konsekvensene av at det faktisk skjer. Basert på disse vurderingene, kategoriseres risikoen som lav, middels eller høy. Deretter utvikles en handlingsplan for å håndtere hver risiko, basert på risikonivået og gruppens evne til å redusere eller eliminere den.

Prosjektgruppen har identifisert flere potensielle risikoer som kan påvirke prosjektets framdrift og suksess. En av de største risikoene identifisert er manglende kompatibilitet mellom ulike verktøy og teknologier som brukes i prosjektet, eller at utviklingen av applikasjonen er mer kompleks enn forutsatt. Dette kan føre til uforutsette forsinkelser og feil som kan påvirke framdriften, og kan føre til at prosjektet krever mer tid og ressurser enn gruppen har til rådighet.

3.5 Evalueringsplan

Gruppen vil først ta for seg de viktigste delene av dette prosjektet, for å bestemme hvilke evalueringer som skal vektlegges mer enn andre. Det er klart at det er et område gruppen vil fokusere mest på for å veie opp om prosjektets mål er oppnådd, ansiktsgjenkjenning og dets effektivitet.

Selv om utviklingen baserer seg på å lage en effektiv løsning i henhold til problemstillingen, ønsker gruppen å gjøre applikasjonen visuelt appellerende og at de mindre funksjonelle egenskapene til applikasjonen skal være velfungerende. Som det å bruke knapper og navigere i applikasjonen, dette refererer til brukergrensesnittet av applikasjonen.

3.5.2 Evalueringsplan av modellen

Da gruppen utvikler en prototype, med formål å fungere som en fremtidig velfungerende programvare med gode funksjonelle egenskaper, anser gruppen det som avgjørende at egenskapene til programvaren oppfyller kravet om presis og effektiv ansiktsgjenkjenning. For å evaluere dette, tenkes det å ta i bruk ytelsesmålinger, som presisjon, nøyaktighet og gjensidig treff.

En grundigere evalueringsmetode og resultater presenteres i kapittel 5.

3.5.3 Evalueringsplan for brukergrensesnitt

For å evaluere brukergrensesnittet av applikasjonen er det ønskelig med en brukertest. En brukertest er nyttig da man kan få innspill og anbefalinger fra andre personer utenfor gruppen. Dette vil hjelpe med å evaluere det funksjonelle aspektet ved brukergrensesnittet til applikasjonen.

For å få klar evaluering av brukergrensesnittet, er det ønskelig å lage en testplan for dette formålet.

4 DETALJERT LØSNING

I dette kapittelet blir metodene, som har blitt anvendt for å løse prosjektet, beskrevet. Her vil det gis en detaljert oversikt over løsningen. I tillegg blir systemets arkitektur og oversikt visuelt beskrevet.

Applikasjonen er utviklet med flere ansiktsgjenkjenningsmodeller, som inkluderer MobileNetV2, ResNetV2 og LBPH. Brukeren har frihet til å velge mellom de ulike modellene som best passer brukerens behov. Implementasjonen av disse modellene i programvaren er et resultat av iterativt arbeid og justeringer gjort under utviklingsprosessen.

Koden i løsningen er strukturert i moduler, noe som gir mulighet for utvidelse av programmet med andre modeller som potensielle brukere ønsker å benytte. En detaljert beskrivelse av modulær koding og gruppens tilnærming til dette vil bli presentert i kapittel 4.3.

Da løsningen inneholder flere modeller, har gruppen valgt å legge vekt og fokus på MobileNetV2 som modell under detaljert løsning og resultater (Kapittel 4 og 5). Denne beslutningen ble tatt med hensyn til at denne modellen gir bedre resultater i lys av prosjektets mål og krav.

Det er verdt å nevne at front-end-komponenten har blitt implementert likt på alle modeller, da de ulike modellene presentert i programvaren i utgangspunktet skal utføre tilnærmet likt arbeid, men på ulik måte.

Struktur for kapittelet

I stedet for å følge en konvensjonell tilnærming der man først presenterer det overordnede bildet av hele applikasjonen, har gruppen gjort et bevisst valg om å starte kapittelet med det "dypeste" laget av applikasjonen, altså ansiktsgjenkjenning og hvordan den er implementert. Deretter vil kapittelet beskrive hvordan denne funksjonaliteten er koblet opp mot front-end-komponenten av applikasjonen, før man til slutt i kapittel 4.3 beskriver sammenhengen mellom de ulike komponentene av applikasjonen.

Valget om å følge denne strukturen baserte seg først og fremst på at gruppen ønsket å legge vekt på den kritiske og komplekse delen av applikasjonen, som er ansiktsgjenkjenning. Ved å starte med en grundig beskrivelse av hvordan denne fungerer, gir kapittelet et solid fundament for forståelsen av applikasjonens kjernefunksjonalitet.

Videre var det ønskelig med en tydelig sammenheng mellom ansiktsgjenkjenningen og front-end-komponenten av applikasjonen. Ved å presentere implementeringsdetaljer tidlig, blir det enklere å forklare hvordan det funksjonelle i back-end-komponenten blir integrert i front-end-komponenten, og hvordan brukergrensesnittet drar nytte av kjernefunksjonaliteten.

Den valgte strukturen for kapittel 4 legger til rette for en logisk progresjon ved å begynne i det innerste laget av applikasjonen og ende opp på den mer abstrakte delen av applikasjonen.

4.1 Back-end

Back-end-komponenten fungerer som en "motor" for applikasjonen, og styrer alt som skjer i bakgrunnen for å sikre at applikasjonen fungerer som den skal. Denne komponenten av løsningen viser til delen som er ansvarlig for å håndtere logikk og data som ikke er synlig for brukeren.

I applikasjonen har det blitt implementert en back-end som står for behandling av live video. Dette gjøres ved hjelp av biblioteket EmguCV. Bilder blir mottatt i form av matriser som blir representert av klassen Mat fra biblioteket EmguCV.

```
private void ImageGrabbed(object? sender, EventArgs e)

{
    Mat frame = new();

    // Retrieves the frame from the source
    _videoCapture.Retrieve(frame);
    if (frame.IsEmpty) return;

    // Processes the frame
    Mat outputFrame = _modelHandler.ProcessFrame(frame);

    // Displays the processed frame
    pictureBox1.Image = outputFrame.ToBitmap();

    try
    {
        detectionListBox.Invoke(UpdateCheckedListBox);
    }
    catch (ObjectDisposedException) { }

}
```

Figur 4.1: Kodesnutt som håndterer mottak av bildestrøm.

Disse bildene blir så bearbeidet av en modell (MobileNetV2), og gjennom en rekke prosesser gjenkjennes personene i bildet.

4.1.1 Modell

Utvikling av modellen

For å oppnå god ytelse med henhold til pålitelighet og presisjon er det viktig at modellene forberedes eller trenes riktig. For å oppnå dette må det samles inn treningsdata i form av bilder av personer man ønsker å kjenne igjen. Arbeidet som gjøres før man kan bruke en modell kan sees på som en pipeline. Første steg ved å samle inn bilder kan gjøres på to måter:

1. Man kan forberede treningsdata ved å ta en rekke bilder av en eller flere personer, for så å organisere disse i tilhørende mapper på den brukte maskinen. For et optimalt resultat er det ønskelig med god kvalitet og variasjon i bildene, altså ulike lyssettinger, vinkler, ansiktsuttrykk og eventuelt briller og hodeplagg.

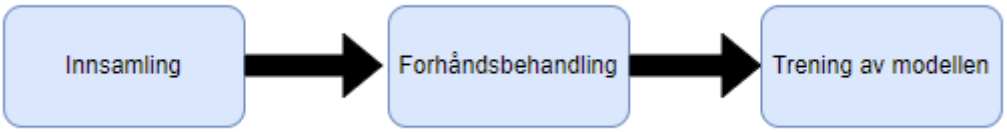
2. En mer realistisk tilnærming for innsamling av treningsdata kan være å benytte verktøy som tilbys av Google eller laste ned bilder manuelt fra nettet. Selv om dette er en mer tidkrevende prosess, oppnår man samme resultat.

Når bildene er samlet inn og organisert i mapper med navn som tilsvarer personenes navn, vil de gå gjennom det andre trinnet i pipelinen. I dette trinnet vil bildene bli behandlet og forberedt for trening av modellen. Denne delen inneholder to steg:

1. Ved å bruke en algoritme kalt Viola Jones og Haar Cascade Classifier, vil ansiktet som befinner seg i bildet bli oppdaget (Tyagi, 2021, avsnitt 1). Dette brukes for å avgrense området hvor ansiktet befinner seg i bildet.

2. Etter at området hvor ansiktet befinner seg har blitt definert, vil bildet skjæres, slik at unødvendig data eller støy fjernes.

Når bildene er ferdig behandlet kan de brukes for å trene modellen. Dette steget vil bli beskrevet mer detaljert i neste kapittel, da dette ikke er nødvendig informasjon for å forstå hvordan pipelinen virker.



Figur 4.2: En visuell representasjon av de overordnede stegene i pipelinen.

Trening av modellen

Etter at bildene er samlet inn og behandlet, blir de lest inn og skalert til en fast størrelse (224x224 piksler) før de konverteres til numeriske verdier mellom 0 og 1. Dette gjøres ved å dele verdiene med 255, som er den høyeste mulige fargeverdien i et bilde.

Når bildene er klare, opprettes en modellarkitektur basert på MobileNetV2, som er en effektiv og kompakt nevral nettverksarkitektur (Keras, u.å.). Modellen blir deretter tilpasset ved å fjerne det øverste laget og legge til nye lag for å spesifisere den til prosjektets oppgave. Den tilpassede modellen består av et GlobalAveragePooling2D-lag, to Dense-lag med 1024 og 512 noder hver, og til slutt et Dense-lag med et antall noder lik antallet unike personer i datasettet. Det siste laget benytter softmax-aktiveringsfunksjon for å produsere sannsynligheter for hver person i datasettet.

For å kunne evaluere modellens ytelse under trening, blir datasettet delt inn i et trenings- og et valideringsdatasett ved hjelp av en 80/20 fordeling. Dette innebærer at 80% av bildene

brukes til å trene modellen, mens de resterende 20% brukes til å vurdere modellens evne til å gjenkjenne personer i bilder den ikke ble trent på. I tillegg til dette vil det være avgjørende å finne de optimale parameterne for modellen, dette vil forklares nærmere i kapittel 5.

Bruk av modellen (recognizing)

Først blir bilder konvertert fra RGB (rød-grønn-blå) til gråskala ved hjelp av EmguCV biblioteket. Deretter blir ansiktene i bildet oppdaget ved hjelp av en rutine som tar inn gråskala bildet og ansiktsdeteksjonsmodellen. Denne modellen kommer i form av en XML-fil som inneholder informasjon om egenskapene som er karakteristiske for ansikter. Denne informasjonen brukes til å lokalisere ansiktene i bildet.

Deretter blir hvert ansikt behandlet for å avgjøre om det er et nytt ansikt eller et som allerede blir sporet. Dette blir gjort ved å sjekke om ansiktet overlapper med ansiktene som allerede er sporet av systemet. Hvis ansiktet allerede blir sporet, blir det ikke gjort noe med det. Men hvis det er et nytt ansikt, blir det opprettet en ny instans av en KCF Tracker fra EmguCV biblioteket, initialisert med ansiktets bounding-box og bildedata. Deretter blir denne trackeren lagt til i systemet, og det aktuelle ansiktet blir tilordnet en etikett ved hjelp av ansiktsgjenkjenningsmodellen.

4.1.2 Visualisering

Ansiktene som blir oppdaget i bildet tegnes på den originale framen til bildet, ved hjelp av en funksjon 'Draw()'. Etikettene blir lagt til ansiktene ved hjelp av 'PutText()'-funksjonen. Dette skjer innenfor en løkke som itererer over alle trackerne som er lagt til i systemet. Hvis sporingen er vellykket blir det tegnet en grønn ramme rundt det gjeldende fjeset. Hvis sporingen mislykkes, blir den tilsvarende rammen og etiketten fjernet fra bildet.

```
for (int i = 0; i < trackers.Count; i++)
            {
                Rectangle trackedFace;
                var success = trackers[i].Update(frame, out trackedFace);
                if (success)
                {
                    // Draws rectangles around each face present
                    frameImage.Draw(trackedFace, new Bgr(Color.Green), 2);

                    // Displays the name of the person
                    CvInvoke.PutText(frameImage, trackedLabels[i],
                        new Point(trackedFace.X, trackedFace.Y - 10),
                        FontFace.HersheySimplex, 0.8,
                        new Bgr(Color.Green).MCvScalar, 2);
                }
                else
                {
                    // Removing labels and trackers
                    trackers.RemoveAt(i);
                    trackedLabels.RemoveAt(i);
                    i--; // Decrement the index to account for removed tracker
                }
            }
```

Figur 4.3: Kodesnutt som håndterer ansiktssporing ved hjelp av KCF Tracker objekter.

Deretter overføres disse oppdaterte bildene til Front-end-komponenten i applikasjonen.

4.2 Front-end

Front-end-komponenten spiller en viktig rolle i brukeropplevelsen og påvirker hvor intuitivt og effektivt brukeren kan navigere i systemet. En godt utviklet front-end kan også være en viktig faktor for å opprettholde brukerens interesse og engasjement i systemet. Det er derfor viktig at det brukes tid på å optimalisere front-end design og funksjonalitet for å sikre en positiv brukeropplevelse.

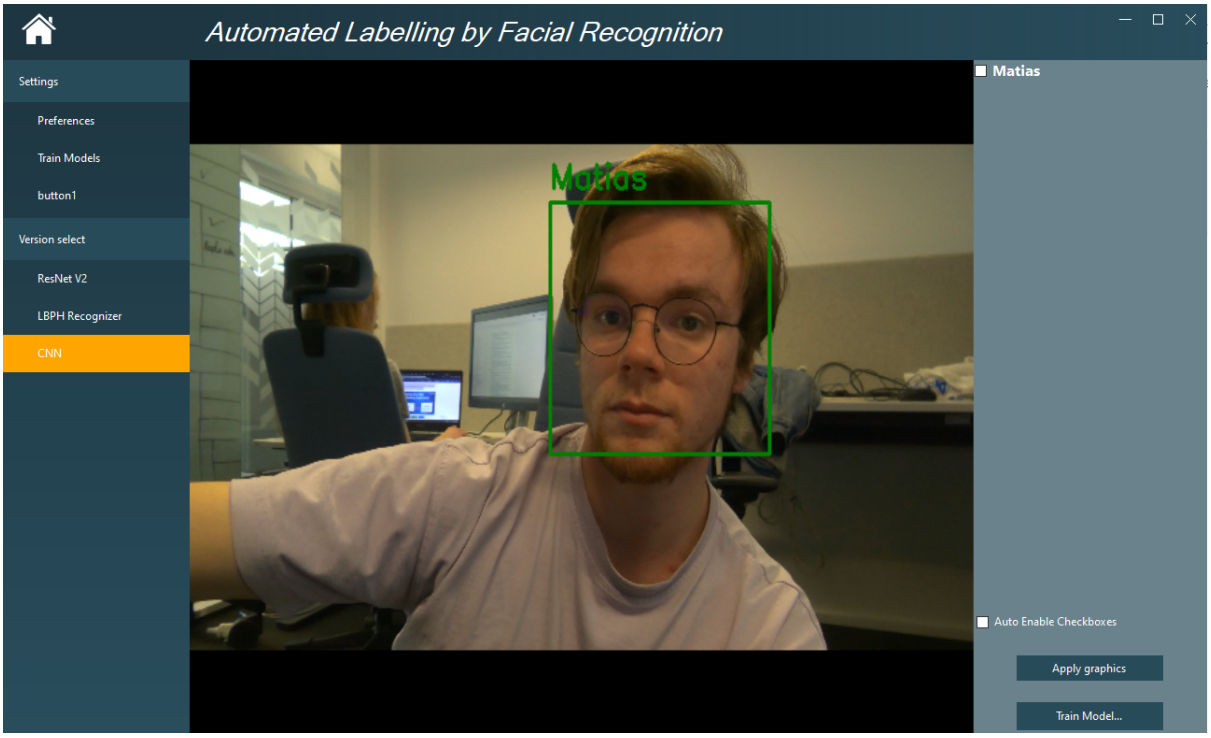
4.2.1 Brukergrensesnitt

Applikasjonen er utformet slik at det skal være enkelt å navigere i den. Det skal være enkelt å bytte modell, og være enkelt å navigere til innstillinger for applikasjonen. Det skal også være

en forside, som man kan navigere til ved å trykke på et hus-ikon øverst i venstre hjørne. Forsiden skal inneholde informasjon om applikasjonen, og hvilke modeller som er implementert.

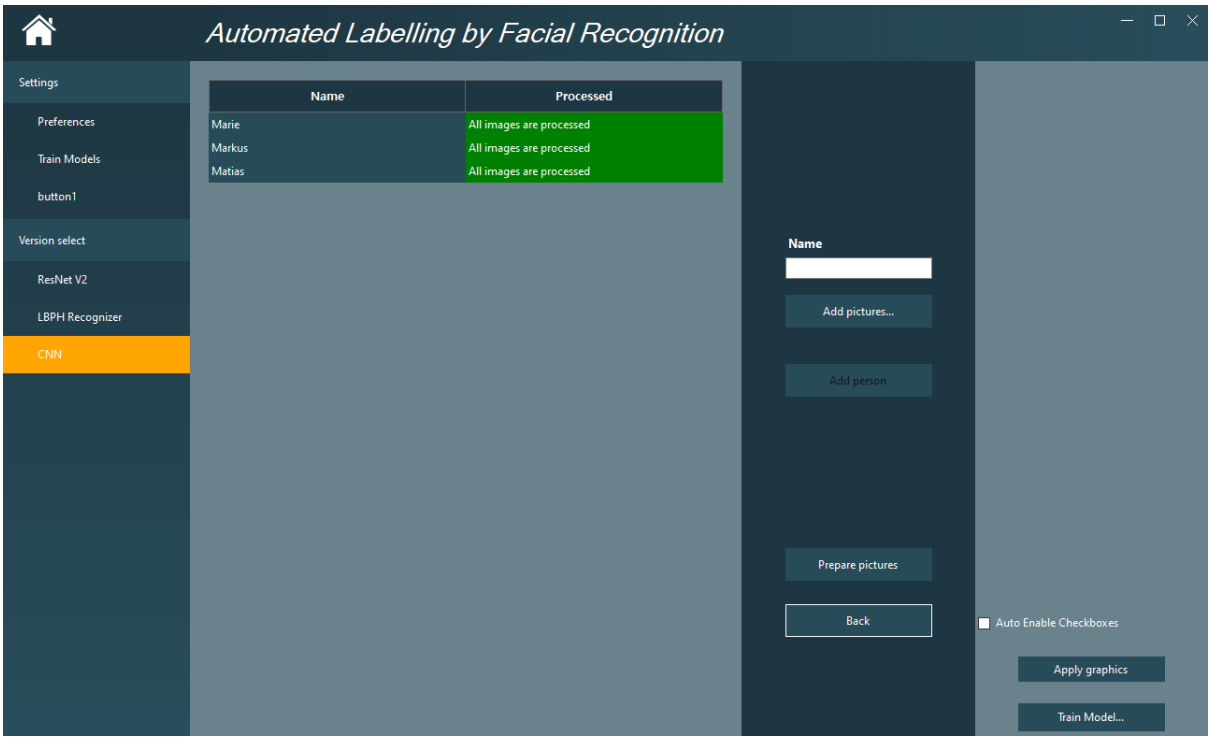
Figur 4.4: Forside for applikasjonen

Når man velger modell, skal den valgte modellen i venstre kolonne lyse oransje for å markere hvilken modell man kjører. Som vist i figur 4.5.



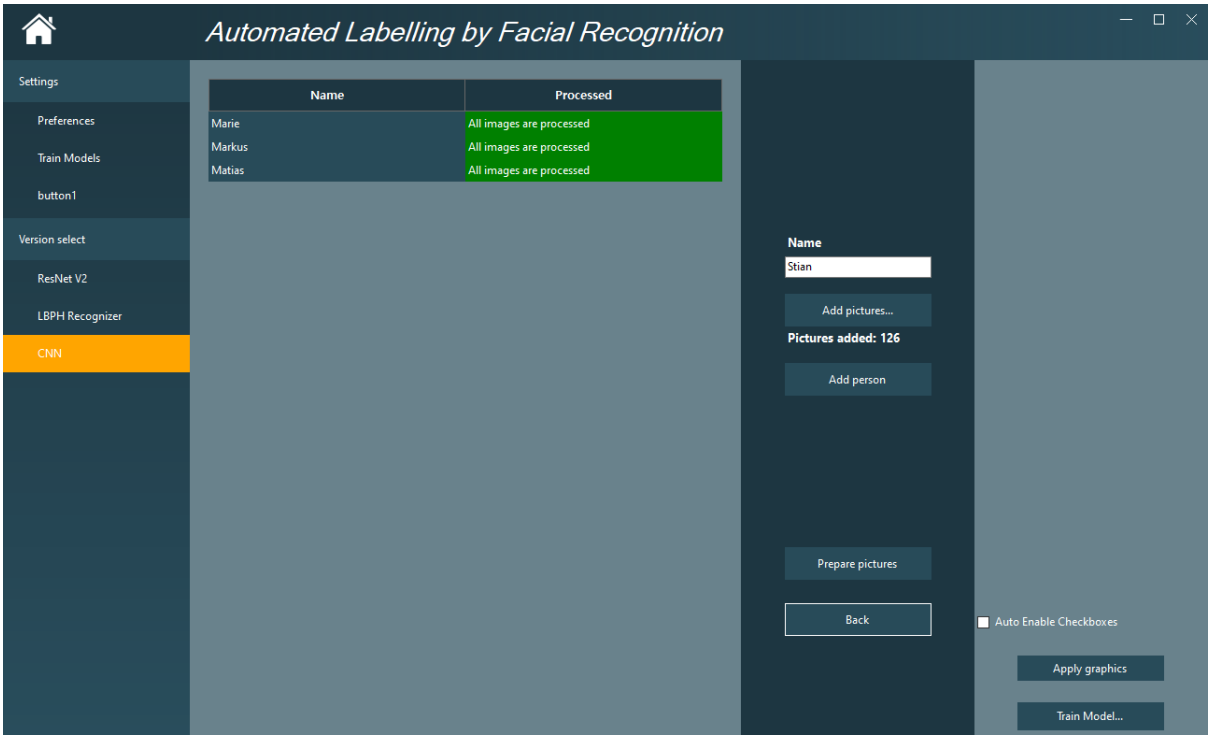
Figur 4.5: brukergrensesnitt for CNN-modellen

Brukergrensesnittet for CNN modellen inneholder knapper for å trene modellen og for å legge på såkalte "lower-third-graphics" for å vise hvem som dukker opp på kamera som vist i figuren over.



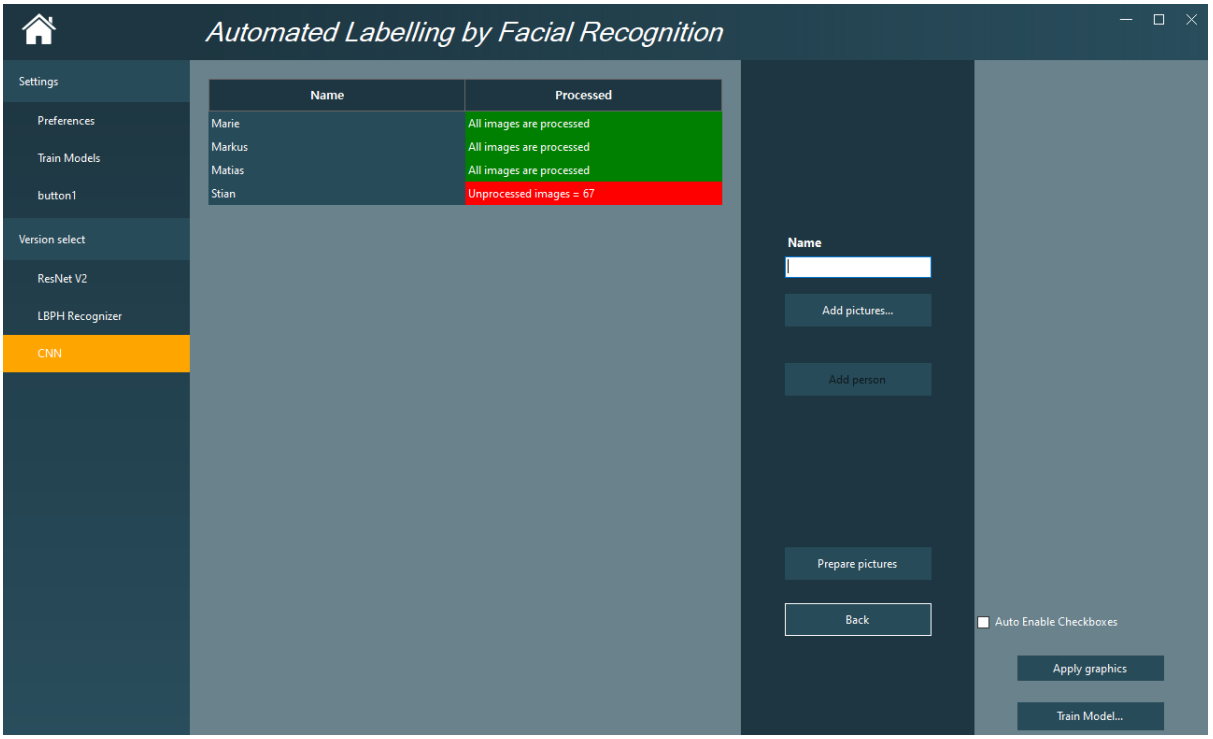
Figur 4.6: Brukergrensesnitt for trening av CNN-modellen.

Brukergrensesnittet for trening av CNN-modellen tilbyr trening på bilder man har av personer på maskinen. Man skriver inn navn under feltet “name”, legger deretter til bilder av den personen, og trykker på “Add person”. “Add person”-knappen er ikke mulig å trykke på før det har blitt lagt til navn og bilder for en person.



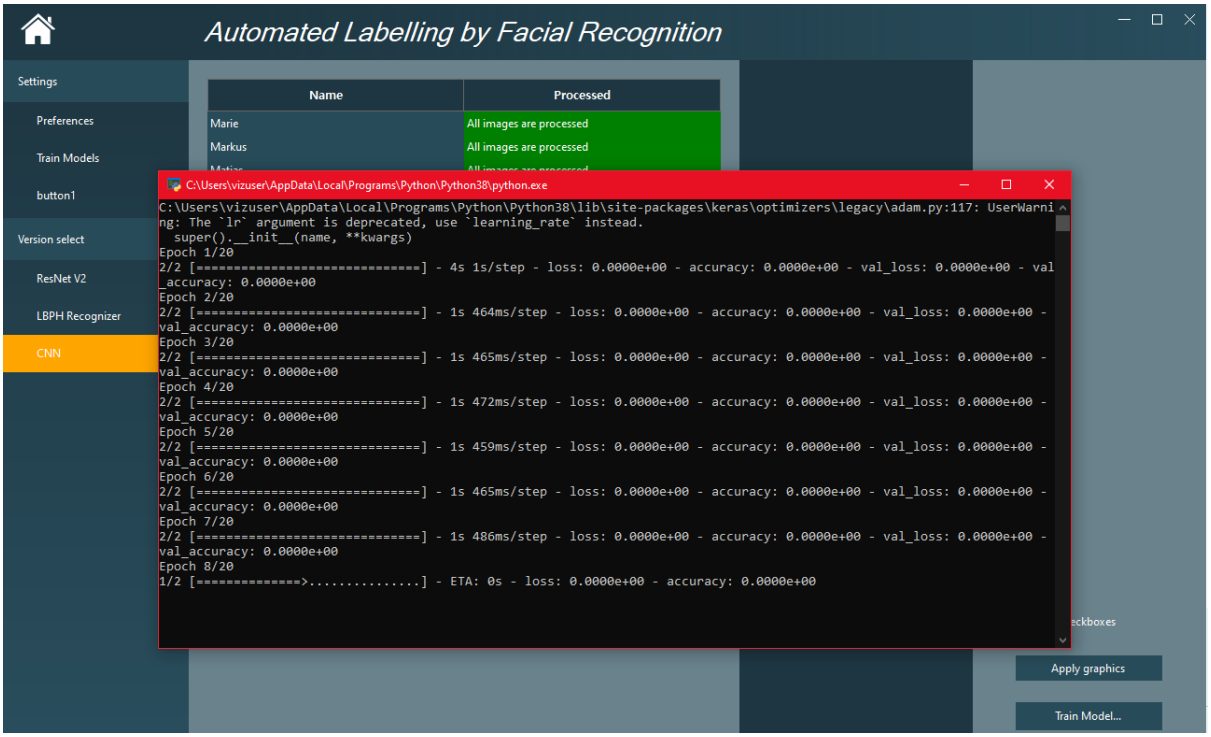
Figur 4.7: Legge til en person med navn Stian, med 126 bilder. CNN-modell.

Når man har lagt til bilder, vises antall bilder som er lagt til og “Add person” knappen blir tilgjengelig som vist i figuren over. Man må deretter legge til personen med de valgte bildene for å trene modellen på disse bildene med det gitte navnet.



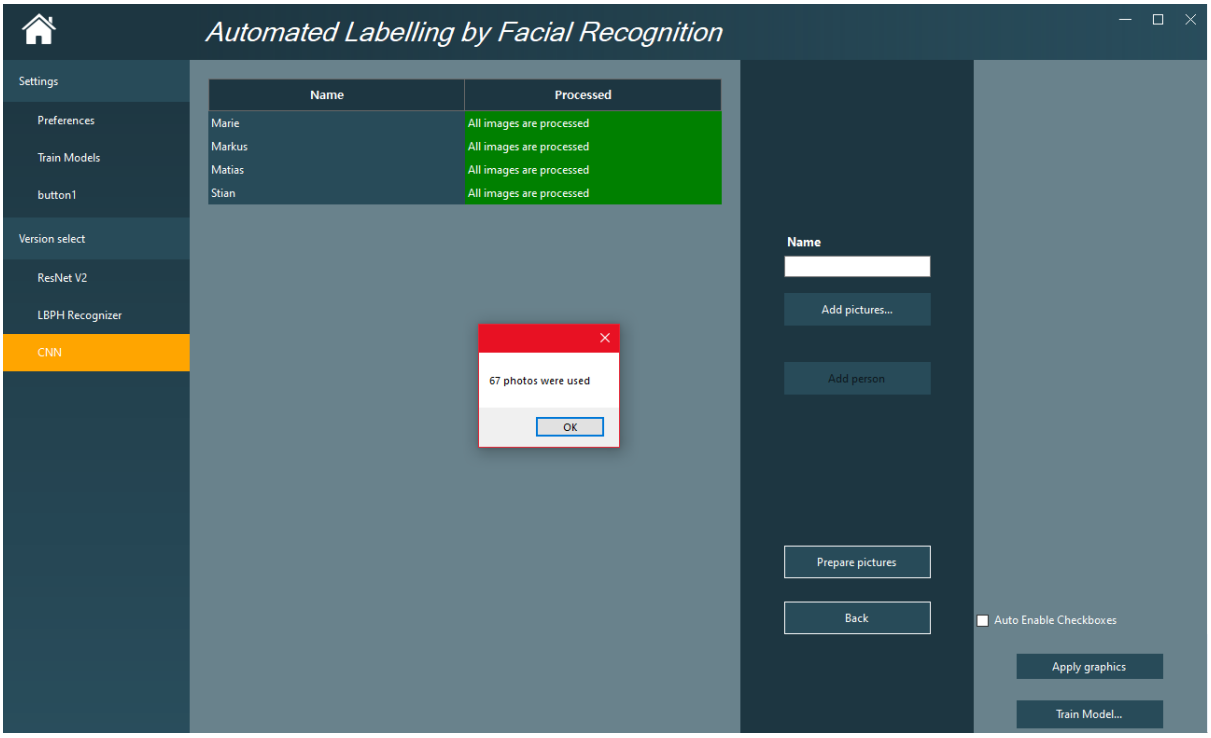
Figur 4.8: Lagt til bilder av en person med navn Stian, og klargjort for trening. CNN-modell.

Når man har lagt til en person, prosesseres bildene så kun de bildene modellen fant et gyldig fjes på blir lagt til og klargjort for trening. I figuren over, viser det at en person med navn Stian er lagt til, og at det er 67 uprosesserte bilder av han. Altså, av 126 bilder, var det kun 67 bilder modellen fant gyldige fjes den kan trenes på. Deretter kan man trene modellen på de uprosesserte bildene ved å trykke på "Prepare Pictures"-knappen.



Figur 4.9: Trening av uprosesserte bilder. CNN-modell

Når treningen er ferdig, viser det at alle bildene er ferdigbehandlet ved at cellen blir grønn for personen under kolonnen “Processed”. Det kommer også opp boks med antall bilder som modellen ble trent på.



Figur 4.10: Trening på de uprosesserte bildene er ferdig, det vises notifikasjon på hvor mange bilder som var med i treningen.

4.2.2 Brukeropplevelse (UX)

Gjennom utvikling av applikasjonen har gruppen fokusert på et enkelt og rent design. Dette er et valg for å bidra til en god og intuitiv brukeropplevelse ved å forhindre at brukeren blir overveldet av kompleksitet og antall valgmuligheter. For å bidra til enkel navigering har gruppen lagt fokus på plassering og tekstinnhold av knappene i programmet. Dette er oppnådd ved bruk av beskrivende tekst, og en naturlig plassering av knappene, slik som rekkefølgen de befinner seg i.

Gruppen har tatt hensyn til fargevalget i applikasjonen, slik at dette er konsistent sammenlignet med merkevaren til prosjektets oppdragsgiver. Ved å bruke de samme fargene som bedriften bruker i sitt merkevarebyggende materiale, kan applikasjonen gi et helhetlig inntrykk av bedriftens identitet og gi en følelse av tilhørighet for brukerne. Dette kan bidra til å styrke bedriftens merkevare og gi en positiv oppfatning av både bedriften og applikasjonen.

Hovedfarger for bedriften

Oransje - #f0824f
Mørkeblå - #1d3641
Hvit - #ffffff

Figur 4.11: Hovedfargepalett for bedriften Vizrt.

4.3 Programsystem, Back-end koblet med Front-end

4.3.1 Modularitet

Gruppen har hatt fokus på modularitet under utviklingen, slik at applikasjonen enkelt kan utvides. En modulær kodebase er en anerkjent og vanlig praksis i moderne programvareutvikling (niharika123, 2018, avsnitt 1). En modulær kode er en praksis der koden er delt inn i uavhengige moduler som hver utfører en spesifikk oppgave eller funksjon. Moduler kan være avhengige av hverandre, men de er isolert fra resten av koden og har definerte grensesnitt for å kommunisere med andre moduler. Dette bidrar til en kode som er enklere å utvikle, teste og vedlikeholde.

Det finnes flere modulære designprinsipp og -mønstre, som for eksempel SOLID-prinsippene og Dependency Injection.

SOLID-prinsippene

SOLID-prinsippene er en samling av fem grunnleggende prinsipper for objektorientert programmering (Erinç, 2020).

1. Single Responsibility (SRP)
   En klasse skal kun ha et ansvarsområde eller grunn til å endre.

2. Open/Closed Principle (OCP)
   Klasser burde være åpne for utvidelse, men lukket for endring. Dette betyr at du kan legge til ny funksjonalitet uten å endre på eksisterende kode.

3. Liskov Substitution Principle (LSP)
   Subtyper skal kunne erstatte sin superklasse uten å endre på applikasjonens korrekthet. Dette vil si at du kan bruke subtyper på samme måte som superklassen uten å bryte applikasjonen.

4. Interface Segregation Principle (ISP)
   Klienter bør ikke tvinges til å avhenge av grensesnitt de ikke bruker. Dette betyr at du bør designe grensesnitt som er spesifikke for klientens behov og ikke inkluderer unødvendige metoder eller funksjoner.

5. Dependency Inversion Principle (DIP)
   Høyere nivåmoduler bør ikke avhenge av lavere nivåmoduler, men heller abstraksjoner. Dette betyr at du bør bruke abstraksjon for å isolere høyere nivåmoduler fra detaljer i lavere nivåmoduler, og gjøre koden mer fleksibel og testbar.

Dependency Injection (DI)

Dependency Injection (DI) er en teknikk for å håndtere ulike avhengigheter mellom deler av koden i et programvaresystem (Karia, 2018). Dette innebærer å flytte ansvar for å opprette og håndtere avhengigheter til en ekstern kilde, slik at klientene ikke behøver å vite om hvordan objektene er satt sammen. Klienten angir kun hvilket avhengigheter den trenger, og injektoren sørger for å gi klienten de riktige avhengighetene.

4.3.2 Lesbarhet

Under utviklingsprosessen har det vært en betydelig innsats for å sikre at koden som er skrevet for programvaren er lesbar. Dette har blitt gjort med hensikt for å forbedre forståelsen og lesbarheten av koden både for gruppemedlemmene og andre utviklere som kan være involvert i prosjektet. Lesbarhet anses som en avgjørende faktor for å oppnå en kvalitetsrik kodebase (Fernandez, 2019, avsnitt 2). Det finnes konvensjoner og praksiser som gjør at koden er mer lesbar, bl. a:

- Navngivning:
  Klasser, funksjoner og variabler burde ha godt beskrivende navn som gjør de forståelige ovenfor deres funksjoner eller hva de representerer. Navn på disse elementene burde være konsistente og følge egne etablerte konvensjoner, for eksempel camelCase eller snake_case (mtrahan41, 2020, avsnitt 2).

- Kommentering
  Der det er nødvendig, burde koden kommenteres for å forklare hva koden er ment for

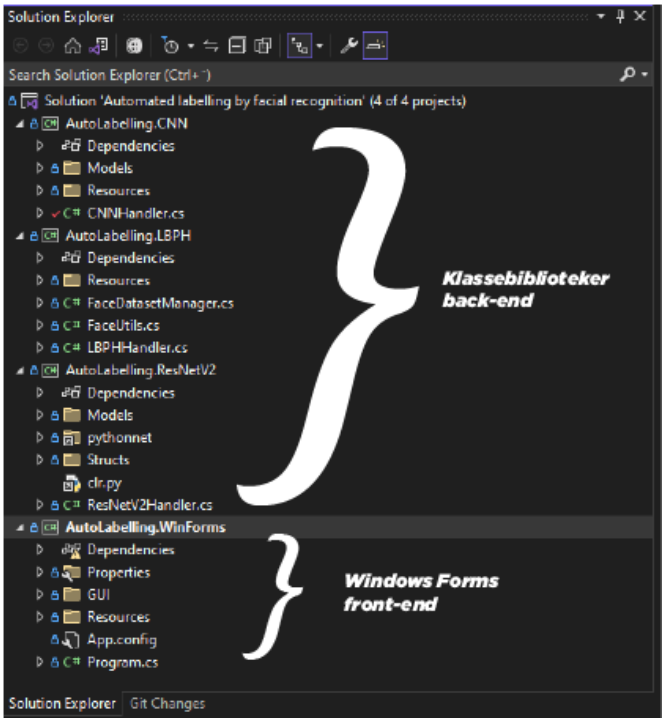
eller hvorfor den er skrevet på en bestemt måte. Dette burde være korte og konkrete kommentarer.

- Kodeoppbygging
  Koden burde være organisert på en måte så den er enkel å navigere. Dette kan inkludere bruk av moduler, funksjoner og klasser for å dele opp funksjonalitet i mindre enheter. Dette bidrar også til å oppnå en modulær kodebase.

4.3.3 Løsning

Det er ønskelig med en kodebase som er modulær for dette prosjektet. Spesielt da programvaren er en prototype som skal kunne testes, utvides og vedlikeholdes. For å kunne oppnå en modulær kode, har det blitt tatt hensyn til SOLID prinsippene og Dependency Injection under utvikling.

Gruppen har sett på det som særdeles viktig å skille front-end-komponenten fra back-end-komponenten for å oppnå modulær kode. Dette er fordi back-end komponenten og front-end komponenten har helt ulike ansvarsområder som beskrevet i kapittel 4.1 og 4.2. Dette viser til SOLID sitt første prinsipp "Single Responsibility", da man kan se på de to komponentene som klasser.



Figur 4.11: Viser oppdelingen av back-end komponenten i form av klassebibliotek og frontend komponenten i form av Windows Forms

I utviklingsprosjektet er back-end komponenten laget som flere klassebibliotek-prosjekter, og front-end delen funker som et presentasjonslag for disse, i dette tilfellet et Windows Forms prosjekt. Oppdelingen vises i figur 4.11. Windows Forms prosjektet tar i bruk klassebibliotekene via prosjekt-referanser. Domenemodellen i vedlegg 3 (Kravdokumentasjon) illustrerer hvordan MobileNetV2 (back-end) komponenten er integrert med front-end komponenten.

På denne måten kan back-end komponenten brukes på nytt i andre prosjekter og på andre plattformer selv om Windows Forms prosjektet er Windows spesifikk.

Under utviklingen har det vært et fokus på å opprettholde en klar separasjon mellom ulike deler av koden. Dette tillater at fremtidige endringer i koden kan gjøres på en klar måte. De ulike klassene er derfor implementert på en slik måte at de kan håndtere endringer, uten å endre eksisterende funksjonalitet. Dette medfører at en klasse blir åpen for utvidelse, men lukket for modifikasjoner. Dette viser til prinsippet "Open/Closed Principle" i SOLID-prinsippene.

Da gruppen utvikler en prototype som skal være mulig å utvide eller endre på i fremtiden, er det viktig at koden er lesbar. Under utvikling, har det blitt tatt hensyn til navngivning av klasser, funksjoner og variabler, slik at de er lett forståelige. Det har også blitt lagt til kort og konsis dokumentasjon for funksjoner og klasser som spiller nøkkelroller i programvaren i form av kommentarer. Disse kommentarene, vil hjelpe en eventuell utvikler å forstå seg på koden og dens funksjonalitet.

```
        /// <summary>
        /// The videoplayer form using the CNN model
        /// </summary>
        public CNNVideoPlayerForm()
        {
            InitializeComponent();
            displayedNames_label.Visible = false;
            _modelHandler = new();

            string? camIndex = ConfigurationManager.AppSettings["DefaultCamera"];
            if (camIndex == null) throw new NullReferenceException();
            _videoCapture = new(int.Parse(camIndex));
            _videoCapture.ImageGrabbed += ImageGrabbed;
            _videoCapture.Start();
        }
```

Figur 4.12: Kodesnutt som håndterer oppretting av vindu for videoavspilling knyttet til MobileNetV2

5 RESULTATER

I dette kapittelet vil resultatene av det omfattende arbeidet som er utført gjennom prosjektet bli presentert.

5.1 Evalueringsmetode

Her tar kapittelet for seg evalueringsmetode for prosjektet, og sikter til evalueringsplanen beskrevet i kapittel 3.5

5.1.1 Evaluering av modellen

For å kunne evaluere og optimalisere modellens ytelse, må den trenes med ulike parametre. For å lette på utfordringene rundt maskinvare, har gruppen brukt kaggle.com for å gjøre evaluering av modellen. Dette er en nettside som tillater utviklere å bygge modeller ved bruk av eksisterende datasett (Uslu, 2022, avsnitt 1). Prosjektet har benyttet seg av deler av VGGFace2 datasettet, som inneholder 3.31 millioner bilder av 9131 personer (Cao *et al.*, 2017, avsnitt 1) for evaluering av modellen.

Resultatene vil bli presentert ved hjelp av ytelsesmålinger som nøyaktighet, presisjon og gjensidig treff. Videre vil resultatene fra de ulike konfigurasjonene og tilnærmingene sammenliknes for å identifisere de mest lovende metodene og parameterne for fremtidig utvikling og optimalisering.

Nøyaktighet

Nøyaktighet (accuracy) er en vanlig metrikk innen maskinlæring for å evaluere ytelsen til en klassifiseringsmodell. Nøyaktigheten representerer forholdet mellom antallet korrekte klassifiserte eksempler og det totale antallet eksempler i datasettet. Nøyaktighet kan uttrykkes som en prosentandel og gir et generelt bilde av hvor godt modellen presterer i å forutsi de riktig klassene (Google, 2022).

Selv om nøyaktighet er en viktig metrikk, så er det ikke tilstrekkelig for å gi en fullstendig evaluering av ansiktsgjenkjenningmodellen, hovedsakelig på grunn av potensielle ubalanser mellom klassene. For dette prosjektet betyr det store forskjeller i antall eksempler per person. Dette kan føre til en skjev modell som favoriserer klasser med flere eksempler og gir en misledende høy nøyaktighet. For å få en bedre forståelse av modellens ytelse i dette tilfellet, er det viktig å vurdere andre metrikker som presisjon og gjensidig treff (recall).

Recall (Gjensidig treff)

Recall er en viktig metrikk som måler modellens evne til å korrekt identifisere positive eksempler blant alle falske positive eksempler. Denne metrikken gir en innsikt i hvor godt modellen klarer å fange opp relevante tilfeller, og er spesielt nyttig i scenarier der det er viktig å minimere antall falske negativer (Google, 2022). Recall er definert som vist i figur 5.1.

I konteksten av ansiktsgjenkjenning kan recall brukes til å vurdere hvor godt modellen identifiserer personer blant alle de faktiske eksemplene av hver person. For eksempel, hvis en modell har en høy recall for en bestemt person, betyr dette at den er i stand til å identifisere et stort antall av de faktiske eksemplene av denne personen. Dette er en viktig metrikk for prosjektets mål, da det er ønskelig å minimere antall falske negativer hvor personer ikke blir gjenkjent.

$$\text{Recall} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

Figur 5.1: Matematisk definisjon av recall

Presisjon

Den siste ytelsesmåling anvendt i prosjektet er presisjon. Presisjonen beskriver hvor presis modellen er med hensyn til antall predikerte positive i forhold til hvor mange faktisk positive det var (Google, 2022). For en matematisk definisjon, se figur 5.2.

Prosjektets oppdragsgiver har ved gjentatte anledninger snakket om viktigheten av å unngå feilaktig navngivning av personer. Dette gjør at presisjon er en svært viktig metrikk, da en høy presisjon betyr at når modellen hevder at et ansikt tilhører en bestemt person, er det stor sannsynlighet for at det riktig.

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

$$= \frac{True\ Positive}{Total\ Predicted\ Positive}$$

Figur 5.2: Matematisk definisjon av presisjon

5.1.2 Evaluering av brukergrensesnittet

For å teste GUI-en (grafisk brukergrensesnitt) til applikasjonen, skal det følges en plan som inkluderer flere steg.

Det vil bemerkes at denne evalueringen ikke er veldig viktig for prosjektets resultat da dette er en prototype egnet for videre arbeid og forbedring. Det at koden er skrevet modulært, betyr at brukere kan velge å bruke back-end-komponenten slik de vil, og ikke avhenge av hvordan brukergrensesnittet er utformet. Derfor er det ikke implementert en detaljert evaluering av brukergrensesnittet. Likevel ønsker gruppen at brukergrensesnittet skal ha en viss standard for funksjonalitet og lager dermed en kort plan for testing som beskrevet under.

Under testing dokumenteres resultatene i form av funn og hvordan GUI-en responderer på ulike interaksjoner. Dette vil hjelpe med å identifisere områder for forbedring og sikre at applikasjonens brukergrensesnitt fungerer optimalt for alle brukere.

Planen inkluderer disse stegene:

1. Identifisere målgruppen
   Det starter med å identifisere målgruppen som skal bruke applikasjonen.
   Dette vil hjelpe med å forstå brukernes behov og krav.

2. Utvikle en testplan
   Lage en plan som beskriver hvordan GUI-en skal testes, hvilke funksjoner som skal testes og hvilke tester som skal utføres.

3. Utføre testingen
   Utfør manuell testing for å verifisere at alle funksjonene lagt til i testplanen fungerer som de skal, og at applikasjonen gir en god brukeropplevelse.

4. Dokumentere resultatene
   Dokumentere resultatene av testingen i form av en rapport som beskriver funnene og anbefalingene for forbedringer.

5.2 Evalueringsresultat

Her presenteres det resultater for evalueringene beskrevet i kapittel 5.1.

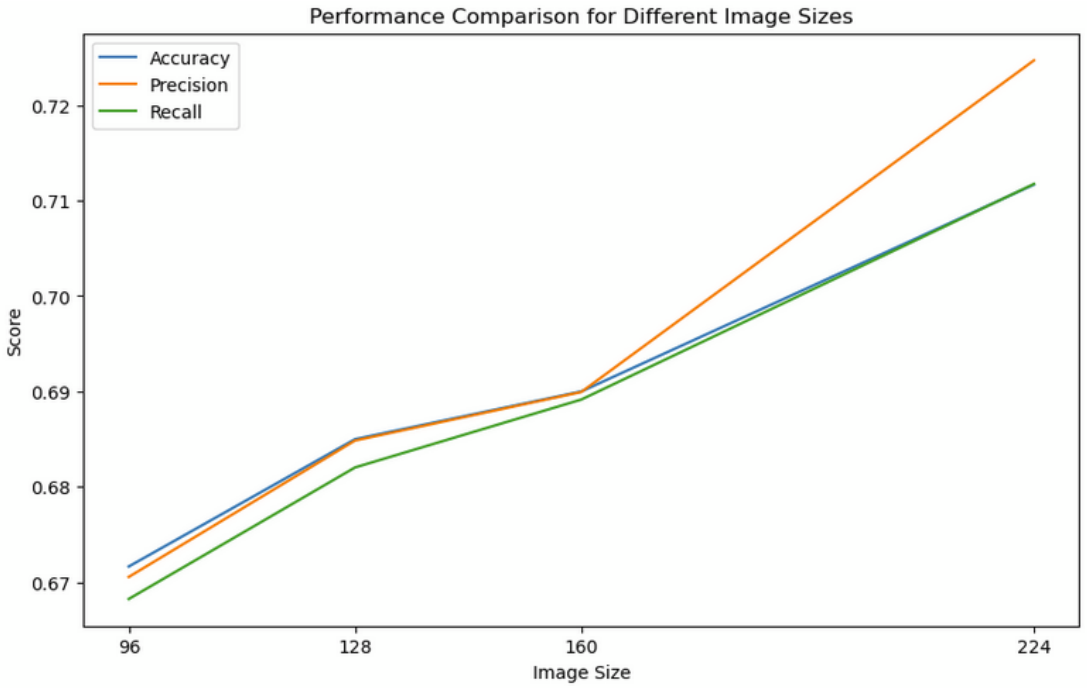
5.2.1 Modell

Modellens ytelse har blitt målt og evaluert i flere stadier, hvor ulikt parameter vektlegges. Optimalt sett, så er det ønskelig å trene modellen på et stort antall personer med mange bilder av hver person, men grunnet manglende datakraft er ikke dette mulig for prosjektet. Ved hjelp av ytelsesmålingene presentert i forrige kapittel, vil resultatene evalueres opp mot hverandre.

Bildestørrelse

De første resultatene viser hvordan størrelsen på bildene påvirker ytelsen til modellen. Når bildestørrelsen økes, øker også antall piksler og dermed mengden informasjon som behandles av modellen. Dette kan føre til økt minnebruk og lengre treningstid, ettersom større bilder krever mer beregningsressurser og mer komplekse modeller for å lære de underliggende mønstrene i dataen. Samtidig kan større bilder inneholde mer detaljert informasjon, noe som potensielt kan forbedre modellens nøyaktighet og evne til å skille mellom ulike personer. Se figur 5.3 for en visuell representasjon av ytelsesmålingene basert på bildestørrelse.

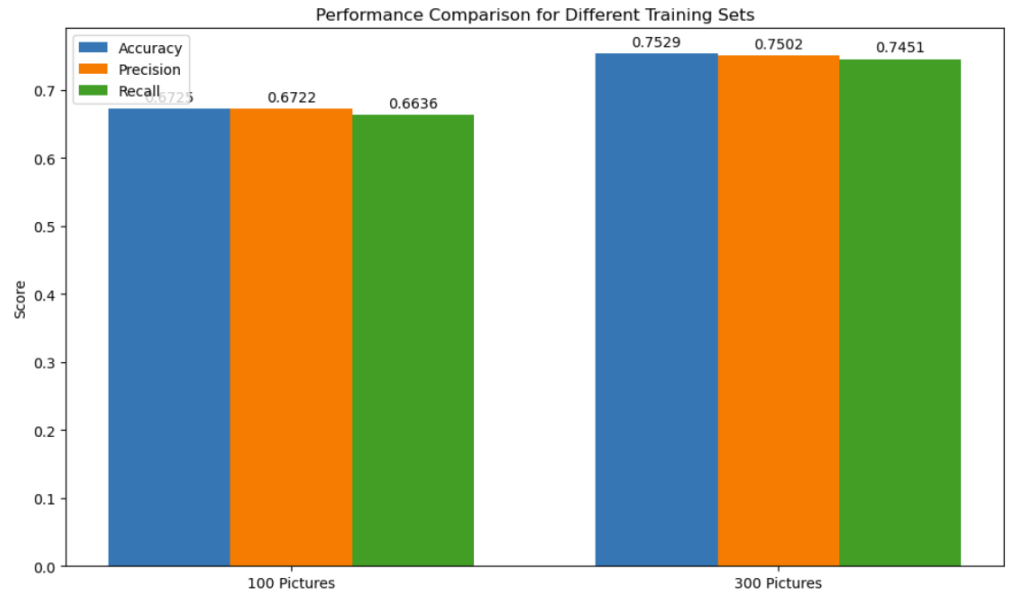
Denne dataen viser at økt bildestørrelse påvirker ytelsen positivt. Ut ifra dette gjøres det en avveining av at en størrelse på 224x224 er et fornuftig valg for dette prosjektet, da en ytterligere økning fører til vansker med hensyn til minnehåndtering og treningstid.



Figur 5.3: Ytelsesmålinger basert på bildestørrelse

Kvantitet

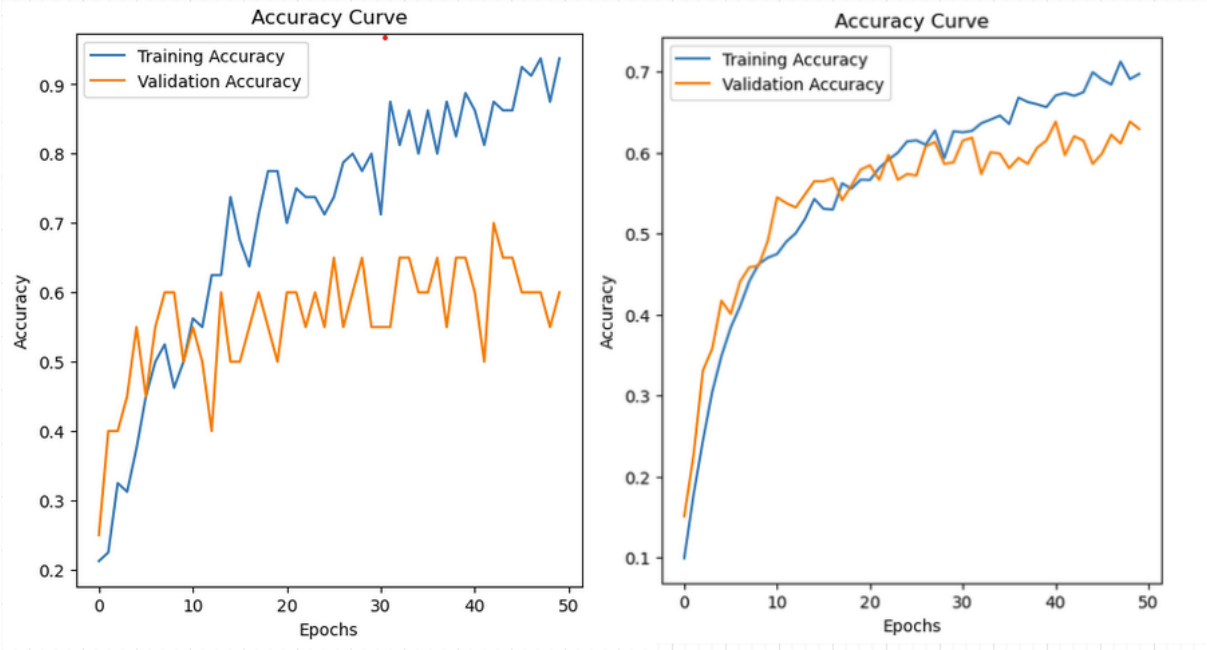
En annen viktig faktor for trening av denne typen modeller er kvaliteten og kvantiteten av treningsdata. Som nevnt tidligere benytter gruppen VGGFace2 som er et sett med god kvalitet og stor mengde data. Prototypen utviklet i dette prosjektet vil kreve tilpasset treningsdata for en eventuell kunde som ønsker å benytte seg av produktet. Det er derfor viktig å vise til statistikk som beskriver hvordan treningsdata påvirker modellens ytelse. Dette innebærer oppløsning, antall personer, antall bilder av hver person og generell kvalitet på bildene. Som nevnt, så er det nødvendig å gjøre visse avgrensninger for å muliggjøre trening av en modell på store mengder data. Resultatene vil derfor ikke gjenspeile den faktisk ytelsen, men i stedet vise forskjellen i ytelse for å representere flere optimale parametre. Figur 5.4 viser hvordan antall bilder per person påvirker ytelsen av modellen. Det er viktig å nevne at denne treningen ble gjort med en bildestørrelse lik 128x128 piksler, noe som gjør resultatet dårligere, men allikevel gjenspeiler viktigheten av antall bilder.



Figur 5.4: Viser hvordan antall bilder påvirker ytelsesmålingene

Overtilpasning

Under trening av en dyplæringmodell, slik som MobileNetV2, er det alltid en risiko for overtilpasning (overfitting). Overfitting kan oppstå dersom modellen utnytter støy og uregelmessigheter som befinner seg i treningsdataen for å utvikle lag (layers) med for høy kompleksitet, som igjen reduserer evnen til å generalisere til ukjent data. Dette kan observeres under trening, ved å sammenligne nøyaktighet på treningsdata og valideringsdata. Dersom modellen presterer godt på treningsdata og stadig dårligere på valideringsdata, tyder dette på overfitting. Det er flere faktorer som kan føre til overfitting, slik som utilstrekkelig antall treningsbilder, kompleksiteten til modellarkitekturen, eller utilstrekkelig regularisering. I dette prosjektet har det blitt anvendt flere teknikker for å forhindre dette, som regularisering, data augmenting, forhåndsbehandling og tidlig stopp. For å oppnå et optimalt resultat med en gitt mengde treningsdata, vil det være nødvendig å justere på flere parametre slik at treningen utføres på en passende måte. Dette innebærer for eksempel lavere treningsmengde (antall epoker) for små datasett og høyere treningsmengde for større datasett.



Figur 5.5: Grafisk representasjon av overtilpasning

Figur 5.5 viser statistikk for to ulike konfigurasjoner, hvor bildet til venstre viser en stor forskjell i validerings og trenings nøyaktighet som er et tegn på overtilpasning. Begge treningene er utført med samme bildestørrelse (224 x 224), 50 epoker, men ulik størrelse i datasettet. Grafen som viser overtilpasning (venstre graf) er et resultat av 5 personer og 20 bilder av hver person, mens grafen til høyre er et resultat av 20 personer og 150 bilder hver. Denne statistikken viser viktigheten av tilstrekkelig treningsdata for å oppnå gode resultater.

Samlet resultat

Statistikken det vises til i dette kapitlet har som hensikt å sammenlikne ytelse mellom forskjellige konfigurasjoner som har blitt testet. Grunnet manglende ressurser har ikke gruppen hatt mulighet til å vise til statistikk med alle de optimale parameterne. Som et resultat av dette er ytelsesmålingene i statistikken litt lavere enn hva de ville vært dersom treningen hadde foregått med optimale parametre. Det er allikevel verdt å påpeke at nøyaktigheten er noe lavere enn ønsket. Det vil være mulig å oppnå bedre resultater ved å finjustere modellen ytterligere, eller eventuelt vurdere andre modeller som potensielt kan yte bedre. Denne problemstillingen vil diskuteres videre i kapittel 6.

5.2.2 Brukergrensesnitt

Evalueringsresultatet for brukergrensesnittet ble utarbeidet av evalueringsmetoden beskrevet i kapittel 5.1.2.

1. Identifisere målgruppen
   For å identifisere målgruppen måtte gruppen ta for seg hvem som faktisk ville bruke applikasjonen. Det var raskt tenkt at bedriften Vizrt, hadde flere kandidater som kunne evaluere brukergrensesnittet da de har utviklet program for kringkastningsindustrien

før, og lyktes med dette (Wikipedia, 2023). Derfor tok gruppen for seg å prate med Vizrt sine ansatte.

2. Utvikle en testplan
   En testplan ble utviklet for å gjennomføre en brukertest av brukergrensesnittet til applikasjonen. Det ble større fokus på det funksjonelle av applikasjonen ovenfor det visuelle, da dette er et verktøy som ikke skal brukes for dets utseende, men hva slags oppgaver den kan utføre.

Testplan		
Funksjoner som skal testes	Funksjoner for vindu	Åpne applikasjonen
		Lukke applikasjonen
		Minimere applikasjonen
		Maksimere applikasjonen (kjøre i fullskjermmodus)
	Innstillinger	Endre kamera for videostrømmen
		Endre modell for ansiktsgjenkjenning
	Ansiktsgjenkjenning	Trene en modell
		Sette navn på personer
Spørsmål som skal stilles	Var knappene intuitive?	
	Var applikasjonen visuelt appellerende?	
	Er det noe som mangler?	

3. Utførelse av testingen
   For å utføre testing av brukergrensesnittet i applikasjonen, fikk gruppen besøk av to ansatte ved Vizrt som gikk lett over applikasjonen og ga tilbakemeldinger om

prototypens brukergrensesnitt. Det var også en kort gjennomgang av funksjonaliteten for programmet for gruppens veileder, Jerry.

4. Dokumentasjon av resultatene
   Resultatene var generelt sett bra. Det var svært lite negative kommentarer til applikasjonen. Men funn og anbefalinger for forbedringer ble skrevet ned.

<table>
<tr><td rowspan="4">Funn</td><td rowspan="2">Ikke alle knapper fungerer som de skal</td><td colspan="2">Train models - knapp</td></tr>
<tr><td colspan="2">Button1 - knapp</td></tr>
<tr><td rowspan="2">Ikke alle modellene er optimalt funksjonelt</td><td>ResNetV2</td><td>Gjenkjenning</td></tr>
<tr><td>LBPH</td><td>Gjenkjenning</td></tr>
<tr><td rowspan="2">Anbefalinger for forbedring</td><td colspan="3">Gjøre forsiden til applikasjonen mer informativ</td></tr>
<tr><td colspan="3">Gjøre trening av modell for CNN modellen mer intuitiv</td></tr>
</table>

Konklusjon

Det var ingen avgjørende funn i testingen av brukergrensesnittet. Knappene som ikke fungerte som de skulle var kun plassholdere til eventuelle fremtidige knapper, og sees dermed ikke på som kritisk. De største problemene som oppstod var funksjonaliteten til de to modellene ResNetV2 og LBPH. Dette var ikke gruppen overrasket over. Da funksjonaliteten til CNN fungerer bra og det er den modellen gruppen har lagt mest vekt på, ser gruppen på den som viktigst. De to andre modellene (ResNetV2 og LBPH) lar gruppen være da de kan utarbeides i fremtiden om det er behov for det.

Alt i alt gikk testene bra, brukerne var fornøyd med designet av programvaren, men påpekte også at det ikke var særlig nødvendig med et visuelt appellerende program som skal utføre et slikt arbeid den er ment for. Det er likevel positivt å ha, ble det også nevnt.

5.3 Prosjektresultat

5.3.1 Funksjonelle og ikke-funksjonelle krav

Funksjonelle krav

Under prosjektet ble det dannet funksjonelle og ikke-funksjonelle krav for applikasjonen. Tabellen under viser funksjonelle krav for applikasjonen som ble planlagt og om de ble fullført.

Nr.	Funksjonelt krav	Planlagt	Fullført
1	Støtte for å vise video fra live kamera.	Ja	Ja
2	Automatisk gjenkjenning av ansikter foran kamera.	Ja	Ja
3	Påskrifter eller merking av gjenkjente personer på skjermen.	Ja	Ja
4	Mulighet for manuell endring av automatisk gjenkjente personer.	Ja	Ja
5	Lagring av historikk over gjenkjente personer.	Ja	Nei
6	Vise antall ansikter som er på kamera.	Ja	Ja
7	Kunne trykke på ansiktets rammer (velge personer i bildet for å gjøre endringer).	Ja	Ja
8	Produktet skal kunne kjøres som en desktop-applikasjon.	Ja	Ja
9	Tilby knapper for pause og starte video.	Ja	Nei
10	Programmet skal kunne aktivere etiketter manuelt og automatisk.	Ja	Delvis

Nr.	Funksjonelt krav	Planlagt	Fullført
11	Integrasjon med eksisterende kringkastingsutstyr og teknologi.	Nei	Nei
12	Mulighet for å legge til eller endre utvalget av ansikter annsiktsgjenkjenningsmodellen trenes på.	Ja	Ja
13	Mulighet for å sette opp integrasjoner med andre systemer.	Ja	Ja

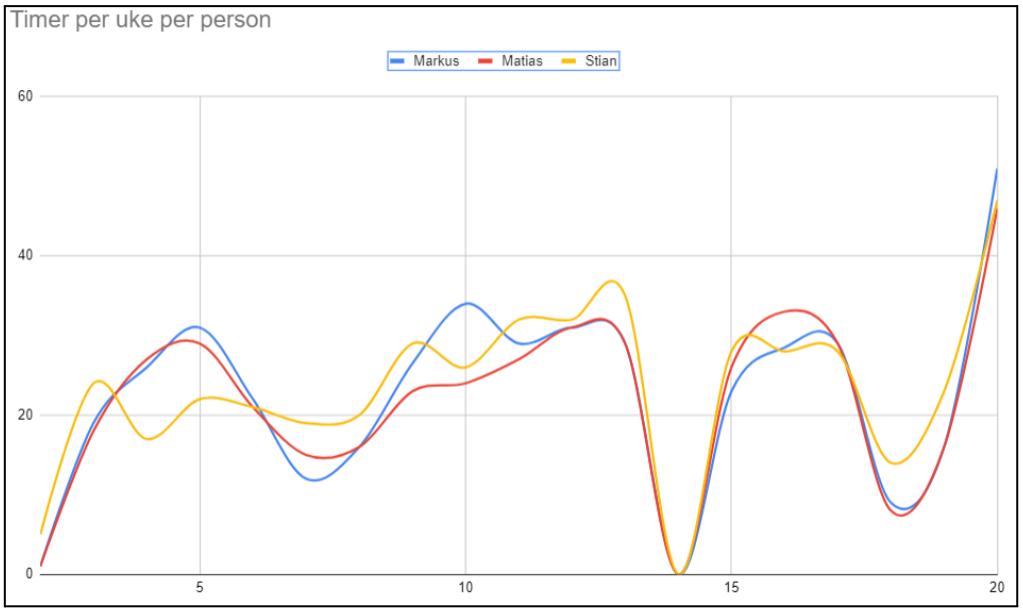
Tabell 5.1: Gjennomføring av funksjonelle krav

Ikke-funksjonelle krav

Ikke-funksjonelle krav er som beskrevet i visjonsdokumentet (Vedlegg 1)

5.4 Prosjektgjennomføring

Et definert og konkret mål for prosjektet var vanskelig å ha helt fra starten av. Gruppen baserte seg på å gjennomføre prosjektet så langt det strakk seg med endring av mål og planer underveis for prosjektet. Fremdriften har vært jevn, men det har vært perioder med mye bruk på utvikling av programvare som ikke var veldig nyttig for prosjektet. Dette viste seg å ikke være et særlig stort problem, da gruppen var godt egnet til å tilpasse seg endringer og lærte av sine feil. Det kan likevel nevnes at denne gjennomføringen ikke alltid er optimal, og vil kreve en del mer arbeid enn nødvendig.



Figur 5.6: Graf over timeliste per person for hver uke

6 DISKUSJON

I dette kapitlet vil det bli gjennomført en analyse av resultatene med hensyn på planer og oppsatte mål. Leseren vil lære om konsekvensene av de valgte tilnærmingene, og hvordan disse har påvirket resultatene som er oppnådd. Diskusjonen vil undersøke hvordan valgene som er tatt, inkludert begrensninger, tilnærminger, strategier og metoder, kan ha påvirket resultatene.

Videre vil dette kapitlet presentere alternative tilnærminger og strategier som kan forbedre resultatene hvis arbeidet skulle gjøres på nytt. Formålet med diskusjonen er å gi en helhetlig forståelse av resultatene og å identifisere områder for fremtidig forskning.

> "Den virkelige kunnskapen er å vite om omfanget av ens egen uvitenhet"
>
> -Konfucius.

6.1 Valg av modell

I løpet av prosjektperioden har flere modeller blitt implementert, hvor utvelgelsesprosessen har involvert en iterativ tilnærming preget av prøving og feiling. Oppdragsgiveren for prosjektet presenterte tidlig deres prototype. Denne prototypen var utviklet i Python, og det ble tydelig at denne hadde utfordringer relatert til hastighet og effektivitet. Som en respons på dette, utviklet gruppen en egen prototype i C#, med mål om å forbedre ytelsen.

MobileNetV2, en lettvektig dyplæringsmodell utviklet av Google, ble valgt for å løse ansiktsgjenkjenningsoppgaven. Modellen er designet for mobile og innebygde applikasjoner, og presterer godt selv med begrensede ressurser (Sandler *et al.*, 2018). Valget av MobileNetV2 er forankret i flere fordelaktige attributter som er vurdert som essensielle for dette prosjektet.

MobileNetV2 er forhåndstrent på ImageNet, et omfattende datasett (ImageNet, 2020), noe som har gjort det mulig å anvende en teknikk kalt transfer-læring. Dette er en teknikk som tillater modellen å dra nytte av allerede lært funksjonalitet, og er spesielt gunstig for dette prosjektet, da det eliminerer behovet for å trene en modell fra bunnen av. Gitt prosjektets tidsbegrensninger, ble det vurdert som upraktisk å utvikle et tilstrekkelig stort datasett for formålet med fullstendig trening fra bunnen av.

Videre, i lys av resultatene presentert i kapittel 5, gir modellens lette og effektive arkitektur en ytelsesforbedring for ansiktsgjenkjenning i sanntid. Til tross for potensiell økning i minnebruk og treningstid, viser resultatene presentert i kapittel 5 at modellen håndterer økt bildestørrelse effektivt. Dette muliggjorde en balanse mellom effektivitet og ytelse, hvor en bildestørrelse på 224x224 viste seg å være et optimalt valg for dette prosjektet.

Resultatene presentert rundt overfitting avdekket at MobileNetV2, selv om den har vist seg effektiv i mange tilfeller, ikke er immun mot denne utfordringen. Det ble anvendt flere

strategier for å forhindre overfitting, som regularisering, data augmentering, forhåndsbehandling og tidlig stopp. Dette har fremhevet behovet for en grundig vurdering av de involverte faktorene, slik som kompleksiteten til modellarkitekturen og størrelsen på treningsdatasettet, for å sikre optimal ytelse. Som presentert i kapittel 5, kan overfitting observeres når modellen presterer godt på treningsdata, men dårligere på valideringsdata. Ved justering av forskjellig parametere, slik som antall trenings-epoker basert på datasettets størrelse, har gruppen forsøkt å minimere overfitting.

Endelig er det viktig å reflektere over modellens generelle ytelse som ble beskrevet i kapittel 5. Resultatene avdekker at, selv om MobileNetV2 har gitt akseptabel ytelse under prosjektets begrensninger, så er den oppnådde nøyaktigheten et nivå lavere enn ønsket. Disse resultatene viser at ytterligere forbedringer gjennom finjustering av modellen, eller vurdering av alternative modeller kan gi bedre resultater.

Ansiktsgjenkjenning i sanntid er en teknisk krevende prosess, og den implementerte modellen virker ikke feilfritt. Dersom prosjektet skulle gjennomføres på nytt, ville en grundigere evaluering av ulike modeller blitt gjort. Det ville også vært ønskelig å ha mulighet til å teste og evaluere ytelsen i et mer realistisk bruksområde enn hva gruppen har gjort. Med et ønske om å levere et godt produkt til oppdragsgiveren innenfor de gitte tidsrammene, har gruppen valgt å beholde MobileNetV2. Samtidig har strukturen til prosjektet blitt designet slik at det vil være enkelt å implementere alternative modeller i fremtiden.

6.2 Valg av programvarearkitektur

Programvarearkitekturen viser til den overordnede strukturen og organiseringen av programvaren til applikasjonen. Det har blitt tatt beslutninger å dele programmet opp i moduler og komponenter som beskrevet i kapittel 4.3. Gruppen mener dette var en viktig faktor for videre utbygging av programvaren da det er en prototype.

Det er også blitt tatt et valg å lage hele applikasjonen som en desktop applikasjon. Dette valget baserte seg på at det var flere erfaringer rundt det å lage en desktop applikasjon innad i gruppen. Dette fungerte godt, da det ville blitt brukt betydelig mer tid på å utvikle en webapplikasjon. For å lage en webapplikasjon, måtte gruppen ha undergått flere timer med å lære seg andre rammeverk enn Windows Forms.

6.3 Utvikling

Utviklingsprosessen i løpet av prosjektet har generelt sett vært problemfri. Det har blitt satt fokus på optimalisering av modellbruken og programvareutvikling til applikasjonen. Dette var to områder gruppen så på som viktig å få til på en god måte for å oppfylle prosjektets funksjonelle krav.

6.3.1 Modellutvikling

Utviklingen av modellen har vært en iterativ prosess, som har gjennomgått flere stadier med testing av ulike konfigurasjoner. Det har vært utfordrende å utvikle og trene modellen for å oppnå et godt resultat.

Prosjektet har benyttet seg av kaggle.com (Kaggle, 2023) for å effektivisere utviklingen, ved at man får tilgang til eksisterende datasett som kan brukes for trening og testing. Dette har bidratt til å lette på deler av arbeidsmengden både for maskinvaren og prosjektmedlemmene.

Prosjektet har, som tidligere påpekt, implementert flere modeller. De initielle iterasjonene hvor MobileNetV2 ble implementert indikerte tydelig at dette var den mest lovende modellen til nå. Modellen ble trent på et sett med bilder av gruppemedlemmene og presterte tilfredsstillende.

Videre i utviklingen var det naturlig å utvide datasettet for å teste modellen ytterligere. I denne delen av utviklingen ble det tydelig at parametriseringen av modellen måtte tilpasses til et større datasett, da prestasjonsmålingene sank sammenliknet med modellen trent tidligere. Med lite erfaring innen ansiktsgjenkjenning fra før, har det vært krevende å komme frem til et godt valg av modell. Noen utfordringer med kompatibilitet mellom Python-modellene og C#-prosjektet har oppstått underveis i prosjektet, og dette har påvirket valg av modell i en viss grad.

6.3.2 Programvareutvikling

Den utviklede prototypen ble en visuell og praktisk måte å demonstrere funksjonaliteten til applikasjonen. Dette har blant annet bidratt til å kunne evaluere ytelsene til de implementerte modellene underveis i utviklingen.

Resultatene fra brukerundersøkelsene viser at gruppen har klart å utvikle et fungerende brukergrensesnitt, som muliggjør enkel bruk og testing av prototypen. Det ble tidlig besluttet å benytte Windows Forms da gruppen hadde kjennskap til dette fra før. Ut ifra resultatene presentert i kapittel 5, vurderes dette som et godt valg.

Den største utfordringen ved applikasjonen er ytelsen til de implementerte modellene. Dette var en utfordring gruppen var klar over. Arkitekturen til programmet har derfor blitt designet slik at andre modeller enkelt kan implementeres.

6.4 Fremgangsmåte

Bruken av Informal Project Management som utviklingsmetodikk har i stor grad vært vellykket for gruppen. Siden prosjektet var av mindre omfang og kun bestod av tre medlemmer, var dette en egnet tilnærming. Da gruppen under hele prosjektet har jobbet tett sammen og brukt verktøy som Google Docs og Discord for å sikre oversikt over fremgang og status, har denne fremgangsmåten vært passende. Det er likevel verdt å diskutere om dette var den optimale løsningen. Hvis gruppen hadde valgt alternative utviklingsmetodikker som

SCRUM eller Kanban, kunne det ha vært enklere å ha faste iterasjoner og definerte versjoner av programvaren, og dermed enklere å holde oversikt over programvareutviklingsprosessen som nevnt i kapittel 6.3.2.

Dersom det hadde blitt valgt en annen fremgangsmåte med strengere rammer, kan det tenkes at gruppen hadde gjort en grundigere undersøkelse for å komme frem til den optimale løsningen. Som en konsekvens av dette har det gjennom prosjektet blitt implementert flere modeller, da kunnskapen til gruppemedlemmene har økt underveis. Selv om en annen metodikk, som SCRUM, kunne bidratt til å forhindre dette, ville det allikevel vært en iterativ prosess da det har vært utfordrende å finne omfattende dokumentasjon om kompatibilitet mellom forskjellig modeller og programmeringspråket C#.

Gruppen vurderer gjennomføringen av prosjektet som god, men med rom for forbedringer. Det ble til tider brukt mye tid på selve programutviklingen på bekostning av å skrive rapporten om utviklingsprosessen. Da programmet fungerer godt, mener ikke gruppen dette nødvendigvis har vært et problem, men dokumentasjonen kunne vært bedre, spesielt i perioder hvor det ble gjort lite dokumentasjon under utviklingen. Dokumentasjon i form av skjermbilder av programmet og notater om programmeringen ville gjort rapporteringen enklere.

7 KONKLUSJON OG VIDERE ARBEID

Dette kapitlet inneholder gruppens konklusjoner basert på gjennomføringen av prosjektet i henhold til problembeskrivelsen. Videre vil det også gis forslag til mulige retninger for videre arbeid basert på resultatene av prosjektet, samt anbefalinger til personer som skal jobbe med lignende prosjekter i fremtiden.

7.1 Konklusjon

For å kunne evaluere hvorvidt prosjektet er vellykket eller ikke, vil det bli gjennomført en sammenligning av prosjektmålene og visjonsdokumentet med resultatene beskrevet i kapittel 5. Prosjektet har hatt et ambisiøst mål om å utvikle en velfungerende prototype for gjenkjenning av personer på direktesendt tv. Grunnlaget for dette målet var et ønske fra oppdragsgiver om å effektivisere denne prosessen, som i dag utføres manuelt. Ved anvendelse av avanserte teknikker innenfor ansiktsgjenkjenning har prosjektet forsøkt å oppnå dette målet, i tillegg til å evaluere prototypens effektivitet og brukervennlighet.

7.1.1 Modell

Evaluering av resultatene viser at prosjektet delvis har nådd de opprinnelige målene. Det har blitt utviklet en funksjonell prototype som kan kjenne igjen personer fra en bildestrøm, gitt at modellen er trent på disse personene. Derimot har utfordringer med nøyaktigheten til modellen på større datasett ført til at løsningen ikke er fullt funksjonell for profesjonelt bruk.

Disse utfordringene er i hovedsak et resultat av begrensninger i tid og ressurser. Et større datasett og mer tid til trening og finjustering av modellen ville vært nødvendig for å oppnå en bedre nøyaktighet. Likevel er resultatene fra prosjektet nyttig, og det er et godt grunnlag for videre utvikling. I tillegg er erfaring og lærdommene fra prosjektet nyttig for videre utvikling og valg av andre modeller.

Modellen dekker til dels de funksjonelle kravene som ble presentert i visjonsdokumentet og gruppen anser utviklingen av prototypen som vellykket, men ikke feilfri. I visjonsdokumentet blir det beskrevet at en modell skal kunne tilordne gjenkjente personer en identitet, noe den gjør. Det blir også beskrevet at etiketten som tilordnes skal være basert på ulike faktorer, som navn, alder eller utdanningsområde. Dette målet er ikke nådd, da etiketten vil bli det samme som navnet på den gjenkjente personen. For å kunne tilpasse etikettene etter ønske, eller for å rette opp i feil, har det blitt lagt til funksjonalitet for dette i brukergrensesnittet.

7.1.2 Brukergrensesnitt

Brukergrensesnittet har blitt til underveis, og var først og fremst en nødvendighet for å kunne teste de implementerte modellene. Windows Forms ble valgt da gruppen hadde erfaring med dette. Det lå ingen krav til grunn for brukergrensesnitt fra oppdragsgiver, dermed ble det besluttet å legge vekt på brukervennlighet og et pent design.

Brukergrensesnittet anses å være fullstendig for prosjektet, og det har blitt lagt til en rekke funksjonaliteter. Brukeren har blant annet mulighet til å velge mellom alle tilgjengelige kameraer for maskinen. Det er også mulig å velge mellom de forskjellige modellene som er implementert. Selv om brukergrensesnittet inneholder det meste av nødvendig funksjonalitet, vil det fortsatt være nødvendig å forberede mapper lokalt på maskinen som inneholder bilder av personene modellen skal trenes på.

Det ble tidlig i prosjektet bestemt en rekke funksjonelle krav som er beskrevet i visjonsdokumentet, og brukergrensesnittet oppfyller de fleste av disse. Slik som automatisk påskrift av etikett, mulighet for manuell endring av etikett og integrasjon med kringkastingsutstyr. Kravet om knapper for pause eller start av video er ikke oppfylt. Dette ble valgt bort av gruppen da applikasjonen i hovedsak skal brukes på direktesendt video. Samlet sett anses denne delen av prosjektet som vellykket.

Det kan likevel nevnes at det ble brukt noe mer tid på det visuelle i applikasjonen enn nødvendig. Da applikasjonen skal brukes som et verktøy for brukere som er ute etter verktøyets funksjonalitet, og ikke dets design.

7.2 Videre arbeid

Dette kapitlet tar for seg anbefalinger for videre arbeid som kan gjøres i prosjektet. Selv om arbeidet har resultert i en fungerende prototype, så er det fortsatt noen aspekter som kan ytterligere forbedres. Før man tar tak i disse anbefalingene, oppfordres det til å gjøre seg kjent med systemdokumentasjonen, vedlegg 4. Dette vil bidra med å gi en bedre forståelse av systemet.

7.2.1 Integrering av FaceNet

Etter omfattende eksperimentering og justering av modellen basert på MobileNetV2, har det kommet frem at en kombinasjon av en modell kalt FaceNet og støttevektormaskin (SVM) potensielt kan yte bedre. Til tross for MobileNetV2 sine sterke sider og ressursbesparende arkitektur, har det vist seg at FaceNet sin nøyaktighet holder seg bedre når størrelsen på datasettet øker.

Det er viktig å bemerke at en fullstendig implementasjon av FaceNet modellen ikke er nådd innenfor prosjektets tidsrammer. Gruppen har i stedet valgt å legge til rette for at modellen enkelt kan implementeres senere. Dersom det viser seg at tiden strekker til, vil en forberedende implementasjon bli forsøkt utført, og finjustering av modellen overlatt til videre arbeid.

7.2.2 Tilpasning av grafikk og visning

Applikasjonen inneholder et vindu hvor videoen vises, og gjenkjente personer tilordnes en navngitt boks som avgrenser område de befinner seg i. Denne grafikken er funksjonell og bidrar til at brukeren enkelt kan overvåke programmet, men den er ikke tilpasset for å vises

på en sending. Videre arbeid vil innebære å bruke informasjon hentet fra applikasjonen til å bestemme innholdet i brukerens egendefinerte grafikk.

7.2.3 Standardisering av treningsdata

Det har ikke blitt utviklet en standardisert prosedyre for å samle inn treningsdata. Det anbefales at videre arbeid inkluderer dette, da det vil bidra til en helhetlig brukeropplevelse, og samtidig minimere sjansen for brukerfeil sammenliknet med den nåværende løsningen. Tidlig i prosjektet hentet gruppen manuelt ned bilder fra nettet og strukturerte disse i tilhørende mapper. Denne prosessen er tidkrevende, og øker sjansen for brukerfeil. En mulig tilnærming kan være å benytte biblioteker fra språk som Python til å automatisere dette.

8 REFERANSER

anandmeg *et.al.* (2023) *Welcome to the Visual Studio IDE | C#.* Tilgjengelig fra: https://learn.microsoft.com/en-us/visualstudio/get-started/csharp/visual-studio-ide (Hentet: 2. Mai 2023)

Boesch G. (2023) *Image Recognition: The Basics and Use Cases (2023 Guide).* Tilgjengelig fra: https://viso.ai/computer-vision/image-recognition/ (Hentet: 21. Mai 2023)

Cao *et al.* (2017) *VGGFace2: A dataset for recognising faces across pose and age.* Tilgjengelig fra: https://paperswithcode.com/paper/vggface2-a-dataset-for-recognising-faces (Hentet: 26. April 2023)

Chaturvedi R. og Verma S. (2022) *Artificial Intelligence-Driven Customer Experience: Overcoming The Challenges.* Tilgjengelig fra: https://cmr.berkeley.edu/2022/03/artificial-intelligence-driven-customer-experience-overcoming-the-challenges/ (Hentet: 2. Mai 2023)

Copper Team (2018) *The Difference between Informal and Formal Project Management Explained.* Tilgjengelig fra: https://www.copperproject.com/2018/03/difference-informal-formal-project-management-explained/ (Hentet: 2. Mai 2023)

Dulčić L. (2020) *Face Recognition with FaceNet and MTCNN.* Tilgjengelig fra: https://arsfutura.com/magazine/face-recognition-with-facenet-and-mtcnn/ (Hentet: 21. Mai 2023)

Erinç Y. (2020) *The SOLID Principles of Object-Oriented Programming Explained in Plain English.* Tilgjengelig fra: https://www.freecodecamp.org/news/solid-principles-explained-in-plain-english/ (Hentet: 2. Mai 2023)

Esler T. (2023) *Face Recognition Using Pytorch.* Tilgjengelig fra: https://github.com/timesler/facenet-pytorch/blob/master/README.md (Hentet: 26. April 2023)

Eurostat (2021) *Artificial intelligence in EU enterprises.* Tilgjengelig fra: https://ec.europa.eu/eurostat/web/products-eurostat-news/-/ddn-20210413-1 (Hentet: 26. April 2023)

Fernandez S. (2019) *Code readability matters.* Tilgjengelig fra: https://www.theguardian.com/info/2019/jan/29/code-readability-matters (Hentet: 9. Mai 2023)

Gentile A. (2020) *How AI is Changing TV Production for the Better.* Tilgjengelig fra: https://www.smpte.org/blog/how-ai-changing-tv-production-better (Hentet: 21. februar 2023)

Google (2022) *Classification: Accuracy.* Tilgjengelig fra: https://developers.google.com/machine-learning/crash-course/classification/accuracy (Hentet: 26. April 2023)

Google (2022) *Classification: Precision and Recall.* Tilgjengelig fra: https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall (Hentet: 26. April 2023)

Google (2023) *Machine Learning Glossary.* Tilgjengelig fra: https://developers.google.com/machine-learning/glossary (Hentet: 21. Mai)

IBM (u.å.) *What is machine learning?* Tilgjengelig fra: https://www.ibm.com/topics/machine-learning (Hentet: 21. Mai 2023)

ImageNet (2020) *ImageNet.* Tilgjengelig fra: https://www.image-net.org/ (Hentet: 19. Mai 2023)

Kaggle (2023) *Kaggle: Your Machine Learning and Data Science Community.* Tilgjengelig fra: https://www.kaggle.com/ (Hentet: 19. Mai 2023)

Karia B. (2018) *A quick intro to Dependency Injection: what it is, and when to use it.* Tilgjengelig fra: https://www.freecodecamp.org/news/a-quick-intro-to-dependency-injection-what-it-is-and-when-to-use-it-7578c84fa88f/ (Hentet: 2. Mai 2023)

Keras (u.å.) *MobileNet, MobileNetV2, and MobileNetV3.* Tilgjengelig fra: https://keras.io/api/applications/mobilenet/ (Hentet: 26. April 2023)

Levine N. (2023) *How to Get ChatGPT to Write Effective Code & Build Websites.* Tilgjengelig fra: https://www.wikihow.com/Get-Chatgpt-to-Write-Code (Hentet: 19. Mai 2023)

Lutkevich B. (2022) *What is the waterfall model?* Tilgjengelig fra: https://www.techtarget.com/searchsoftwarequality/definition/waterfall-model (Hentet: 21. Mai 2023)

McCain A. (2022) *25 Artificial Intelligence Statistics [2023]: Key Facts About AI And The AI Industry [2023].* Tilgjengelig fra: https://www.zippia.com/advice/artificial-intelligence-statistics/ (Hentet: 21. februar 2023)

mtrahan41 (2020) *Coding best practices.* Tilgjengelig fra: https://curc.readthedocs.io/en/latest/programming/coding-best-practices.html (Hentet: 9. Mai 2023)

niharika123 (2018) *Modular Approach in Programming.* Tilgjengelig fra: https://www.geeksforgeeks.org/modular-approach-in-programming/ (Hentet: 26. April 2023)

OliaG *et.al.* (2022) *Why modern desktop applications.* Tilgjengelig fra: https://learn.microsoft.com/en-us/dotnet/architecture/modernize-desktop/why-modern-applications (Hentet: 2. Mai 2023)

OpenCV (u.å.) *About.* Tilgjengelig fra: https://opencv.org/about/ (Hentet: 26. April 2023)

Ortiz P. (2022) *Desktop Application Vs. Web Application: What's the difference?* Tilgjengelig fra: https://flexisourceit.com.au/resources/blog/information-technology/desktop-application-vs-web-application-whats-the-difference/ (Hentet: 2. Mai 2023)

Q.ai (2023) *Applications of Artificial Intelligence Across Various Industries.* Tilgjengelig fra: https://www.forbes.com/sites/qai/2023/01/06/applications-of-artificial-intelligence/ (Hentet: 26. April 2023)

Rana K. (2021) *Difference between Web Application and Desktop Application.* Tilgjengelig fra: https://artoftesting.com/difference-between-web-application-and-desktop-application (Hentet: 19. Mai 2023)

Salton do Prado K. (2017) *Face Recognition: Understanding LBPH Algorithm.* Tilgjengelig fra: https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b (Hentet: 21. Mai 2023)

Sandler *et al.* (2018) *MobileNetV2.* Tilgjengelig fra: https://paperswithcode.com/method/mobilenetv2 (Hentet: 19. Mai 2023)

Sokolova E. (2021) *SCRUM flexible methodology — what is it and why you need it.* Tilgjengelig fra: https://startupjedi.vc/content/scrum-flexible-methodology-what-it-and-why-you-need-it (Hentet: 2. Mai 2023)

Torsvik A. (u.å.) *5 ways Artificial Intelligence and Machine Learning will make sports broadcasting smarter.* Tilgjengelig fra: https://www.vizrt.com/community/blog/5-ways-artificial-intelligence-and-machine-learning-will-make-sports-broadcasting-smarter (Hentet: 21. februar 2023)

Tyagi M. (2021) *Viola Jones Algorithm and Haar Cascade Classifier.* Tilgjengelig fra: https://towardsdatascience.com/viola-jones-algorithm-and-haar-cascade-classifier-ee3bfb19f7d8 (Hentet: 2. Mai 2023)

Uslu Ç. (2022) *What is Kaggle?* Tilgjengelig fra: https://www.datacamp.com/blog/what-is-kaggle (Hentet: 26. April 2023)

Vizrt (u.å.) *About us.* Tilgjengelig fra: https://www.vizrt.com/vizrt (Hentet: 7. mars 2023)

Walther-Zhang Y. (2021) *1 av 10 foretak bruker kunstig intelligens-teknologi.* Tilgjengelig fra: https://www.ssb.no/teknologi-og-innovasjon/informasjons-og-kommunikasjonsteknologi-ikt/statistikk/bruk-av-ikt-i-naeringslivet/artikler/1-av-10-foretak-bruker-kunstig-intelligens-teknologi (Hentet: 21. februar 2023)

Wikipedia (2023) *Kanban (development).* Tilgjengelig fra: https://en.wikipedia.org/wiki/Kanban_(development) (Hentet: 21. Mai 2023)

Wikipedia (2023) *Vizrt.* Tilgjengelig fra: https://en.wikipedia.org/wiki/Vizrt (Hentet: 19. Mai 2023)

9 VEDLEGG

- Vedlegg 1: Visjonsdokument.
- Vedlegg 2: Prosjekthåndbok.
- Vedlegg 3: Kravdokumentasjon.
- Vedlegg 4: Systemdokumentasjon.

10 ORDLISTE

ORD	BESKRIVELSE
Algoritme	En serie av instruksjoner som utføres av en datamaskin for å løse et problem eller utføre en oppgave.
Ansiktsgjenkjenning	Teknologi som brukes til å identifisere en person basert på deres ansiktstrekk.
Back-end	Den delen av en nettside eller applikasjon som håndterer databehandling, logikk og lagring av data, og som vanligvis ikke er synlig for brukeren.
Bounding boxes / avgrensingsbokser	Refererer til en rektangulær boks som brukes til å definere og avgrense et objekt eller et område i et bilde eller en video.
Bug	Refererer til en feil eller en uventet oppførsel i en programvare eller en applikasjon.
C#	Et objektorientert programmeringsspråk.
CPU - Central Processing Unit	Hjernen til en datamaskin, den utfører de grunnleggende instruksjonene som trengs for å kjøre programvare og håndtere data.
CUDA	(Compute Unified Device Architecture) er en teknologi som gjør det mulig for datamaskiner å utføre parallell databehandling ved hjelp av en GPU.
Desktop Applikasjon	En programvareapplikasjon som er designet for å kjøre på en datamaskin, i motsetning til en webapplikasjon som kjører i en nettleser.
Deteksjon	Refererer til prosessen med å finne og identifisere objekter eller mønstre i et bilde eller en video.

ORD	BESKRIVELSE
Dyp læring	En type maskinlæringsalgoritme som bruker flere lag med kunstige nevrale nettverk til å lære og forbedre seg selv basert på erfaring fra store datasett.
Embedding	Refererer til en teknikk i maskinlæring som involverer å representere data, for eksempel ord eller bilder, som vektorer i et høydimensjonalt rom.
EmguCV	Åpent kildekode-bibliotek for bildebehandling og evnen til datamaskiner til å forstå og tolke visuell informasjon.
Face-Tracking	Refererer til prosessen med å følge en persons ansikt i en video eller en serie av bilder.
Front-end	Front-end refererer til den delen av en nettside eller applikasjon som brukeren interagerer med direkte, inkludert grensesnittet, brukeropplevelsen og visuelle designelementer.
Git	En distribuert versjonskontrollsystem som brukes av utviklere for å håndtere endringer i kildekode og samarbeide på prosjekter.
GitLab	GitLab er en web-basert plattform for kildekodehåndtering, som gir utviklere og team muligheten til å samarbeide om utvikling av programvare og prosjekter.
GPU - Graphics Processing Unit	En spesialisert prosessor som er utviklet for å håndtere grafisk databehandling, for eksempel rendering av bilder og videoer.
IDE - Integrated Development Environment	En programvareapplikasjon som gir en komplett plattform for å utvikle, teste og feilsøke programvare.

ORD	BESKRIVELSE
KCF Tracker	En algoritme innenfor bildebehandling som brukes til å spore og følge objekter i en videosekvens basert på visuelle egenskaper.
KI - Kunstig Intelligens	Refererer til en teknologi som gjør det mulig for maskiner å utføre oppgaver som vanligvis krever menneskelig intelligens, som for eksempel å lære, planlegge, resonnere, gjenkjenne og se.
Klassifiseringsmodell	En modell innenfor maskinlæring og kunstig intelligens som lærer å kategorisere eller gruppere data basert på tidligere trening og mønstergjenkjenning, og brukes til å forutsi kategorien til nye, ukjente data.
Kildekode	Refererer til de skriftlige instruksjonene som en utvikler skriver for å lage programvare.
ML - Maskinlæring	En gren innenfor kunstig intelligens som innebærer utvikling av algoritmer og modeller som kan lære og forbedre seg selv basert på erfaring fra dataene de behandler.
Modell (KI)	En KI-modell er en algoritme eller et system som er trent på et datasett for å utføre spesifikke oppgaver innenfor kunstig intelligens.
OpenCV	(Open Source Computer Vision Library) er et åpen kildekode-bibliotek for bilde- og videobehandling.
Pipeline	En sekvens av steg eller prosesser som behandler data i en bestemt rekkefølge for å oppnå ønsket resultat.
Prototype	En tidlig versjon av et produkt eller en løsning som brukes til å teste konseptet og identifisere forbedringsmuligheter.

ORD	BESKRIVELSE
Python	Et høynivå programmeringsspråk som er populært i data science, maskinlæring og kunstig intelligens.
Treningsdata	Refererer til de dataene som brukes til å trene en maskinlæringsmodell.
Visual Studio	Et integrert utviklingsmiljø (IDE) som brukes til å utvikle programvare for Windows-plattformen.