Sigurd Løite Jacobsen
Eivind Alfsvåg Johansen

Bacheloroppgave

Tegnspråk med maskinlæring

Mai 2020

NTNU
Norges teknisk-naturvitenskapelige universitet.
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for datateknologi og informatikk

NTNU
Kunnskap for en bedre verden

Bacheloroppgave

2020

NTNU

Kunnskap for en bedre verden

Sigurd Løite Jacobsen
Eivind Alfsvåg Johansen

Tegnspråk med maskinlæring

Bacheloroppgave
Mai 2020

NTNU
Norges teknisk-naturvitenskapelige universitet.
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for datateknologi og informatikk

NTNU
Kunnskap for en bedre verden

Forord

Grunnen til at denne oppgaven ble valgt er fordi den virket både spennende og utfordrende, samtidig fant vi begge interesse i å finne ut hvilket sluttprodukt det ville være mulig å oppnå. Sluttproduktet vil også kunne hjelpe folk å bli forstått. Det er ikke er laget noen god løsning på denne oppgaven fra før. Det var også områder som vi hadde liten eller ingen kunnskap om, som gjorde dette prosjektet enda mer spennende. Vi så derfor at vi kunne lære mye av denne oppgaven.

I denne oppgaven utforsket vi måter å gjenkjenne tegnspråk på. Vi testet og sammenlignet ulike teknologier og maskinlæringsmetoder, og sjekket hva som egnet seg best for denne typen oppgave. Sluttproduktet vårt består av maskinlæringsalgoritmer som oppnår gode resultater på gjenkjenning av hele det ASL-alfabetet ved bruk av Openpose keypoints og videoer fanget av et rgb-kamera.

Vi vil takke vår veileder Tomas Holt og oppgavegiver 3D Motion Technologies for god veiledning og kommunikasjon gjennom hele prosjektet. Vi vil også takke Anne Malene Wassmo for å ha tatt seg tid til å gjennomføre et intervju om tegnspråk.

Oppgavetekst

Det er mange personer som er døve eller har nedsatt hørsel og som kun klarer å kommunisere ved hjelp av tegnspråk. Det finnes dessverre veldig få som kan tegnspråk, og dette resulterer i at det er mange som ikke klarer å kommunisere med folk rundt seg på en god og forståelig måte.[1] Ønsket med oppgaven er derfor på sikt å kunne lage et system som kan ta opp tegnspråk, visualisere tegnspråk på en avatar og oversette tegnspråket. Det er også ønskelig å sette avataren inn i en virtuell virkelighet og gi tilbakemelding på om utførelsen er korrekt.

Systemet som skal lages må kunne fange opp bevegelser via ett eller flere bevegelsesfangstsystemer, som for eksempel Leap Motion, Kinect eller andre ferdiglagde systemer. For å gjenkjenne dataen er det naturlig å bruke maskinlæring, så det vil være aktuelt å produsere data selv som passer de ulike bevegelsesfangstsystemene.
Hovedhensikten med oppgaven er å forske på og/eller lage en delløsning for opplæring og oversetting av tegnspråk. Oppgaven er for stor til å gjennomføres som et bachelorprosjekt, så hovedfokuset ble rettet mot maskinlærings- og bevegelsesfangstdelen av oppgaven.

Senere i prosjektet ble oppgaven rettet mot å undersøke ulike teknologier for bevegelsesfangst for å finne ut hvilke som egner seg best til gjenkjenning av tegnspråk. Det ble også jobbet med å teste hvilke maskinlæringsmetoder som passer best til dataen som ble laget med valgt sporingsteknologi.

Sammendrag

Målet med denne bacheloroppgaven Tegnspråk med Maskinlæring er å kunne oversette tegnspråk ved hjelp av maskinlæring. Det er flere teknologier som må kunne jobbe sammen for å kunne oppnå et bra resultat, blant annet maskinlæringsmodeller og sporingsteknologier. Et av hovedfokusene var å finne en sporingsteknologi som kan spore nøyaktig og er enkel å bruke. Det ble sett på mange ulike områder innen maskinlæring for å se etter ideer som kan virke positivt på prosjektet. I tillegg ble metoder for databehandling testet for å finne gode måter å strukturere data på og oppnå tilfredsstillende resultater. Resultatene som blir beskrevet i rapporten ble oppnådd ved hjelp av databehandlingsmetoder og mye testing og justering av forskjellige maskinlæringsmodeller.

Datasettet brukt på maskinlæringsmodellene inneholder både statiske og dynamiske håndbevegelser, noe som var utfordrende å oversette i starten, men ble oppnådd mot slutten av prosjektet. Det ble muliggjort av databehandlingsmetodene som økte nøyaktigheten på modellene. Resultatene viser at oversetting av tegnspråk ved hjelp av maskinlæring er mulig, selv med av todimensjonale koordinater.

Resumé

The purpose of this bachelor thesis, Sign language with Machine Learning, is to be able to translate sign language using machine learning. Technologies that must work together to achieve a good result, are machine learning models and tracking technologies. One of the main focuses was to find a tracking technology that can track accurately and is simple to use. A lot of different concepts of machine learning were studied so that good ideas that could positively affect the project would be implemented. Additionally methods for data processing were tested to find good ways of structuring data and to achieve satisfying results. The results presented in this report were achieved by using data processing methods and testing and adjusting different machine learning models.

The dataset used on the machine learning models both contained static and dynamic gestures, which was challenging to translate in the beginning of the project, but was accomplished towards the end. This was also achieved by using data processing methods which increased the accuracy of the models. The results shows that translation of sign language using machine learning is possible even using two dimensional coordinates.

Innholdsfortegnelse

Kapittel 1: Introduksjon og relevans

Kommunikasjon muliggjør deling av følelser, tanker og meninger mellom mennesker. Dette binder oss sammen og har ført til at menneskeheten har utviklet seg mye siden den oppstod, og den kan fortsette å utvikle seg. Kommunikasjon mellom mennesker foregår som regel muntlig, men det finnes også de som ikke har mulighet til å snakke eller høre. I sosiale sammenhenger har de døve og stumme bruk for en tegnspråkstolk for å kommunisere med andre. Dette kan skape vanskeligheter hvis det må utveksles sensitiv informasjon eller hvis det skulle oppstå en nødssituasjon. Som en konsekvens av dette føler mange døve og stumme seg isolert fra samfunnet.[2] Dette var en stor motivasjonsfaktor for valget av oppgaven.

Gjenkjenning eller oversetting av tegnspråk en veldig utfordrende oppgave da det innebærer tolkning mellom visuell og språklig informasjon. Den visuelle delen består av flere deler som blant annet kroppsbevegelse og ansiktsuttrykk. Å tolke samlingen av den visuelle informasjonen som språklig informasjon er den vanskeligste utfordringen når det kommer til å realisere ønsket om oversettingen av tegnspråk. Denne oppgaven er derfor mer fokusert på oversetting av statiske tegn og noen dynamiske tegn. Da dette hovedsakelig er et forskningsprosjekt er det blitt undersøkt hvilke teknologier som funker best når det kommer til bevegelsesfangst og maskinlæringsmetoder slik at det kan bygges videre på.

Rapporten starter med et teorikapittel hvor det skrives om tegnspråk og maskinlæring. Hensikten med dette kapittelet er å gi nok forståelse for å kunne forstå de neste kapitlene. Videre beskrives det hvordan oppgaven ble utført med tanke på valg av teknologi, databehandling og maskinlæring. Til slutt presenteres resultatene med tabeller og grafer som drøftes mot slutten.

Akronymer og forkortelser

- ASL - American Sign Language, Det dominerende tegnspråket som brukes i USA og deler av Canada. Beskrives i kapittel 2.1.1
- ANN - Artificial Neural Network. Et type nevralt nettverk. Beskrives i kapittel 2.2.2
- RNN - Recurrent Neural Network. Et type nevralt nettverk. Beskrives i kapittel 2.2.2
- CNN - Convolutional Neural Network. Et type nevralt nettverk. Beskrives i kapittel 2.2.2
- TCN - Temporal Convolutional Network. Et type nevralt nettverk. Beskrives i kapittel 2.2.2
- LSTM - Long Short-Term Memory. Et type lag i RNN-nettverk. Beskrives i kapittel 2.2.2
- GRU - Gated Recurrent Unit. Et type lag i RNN-nettverk. Beskrives i kapittel 2.2.2
- RGB - Red, Green, Blue. Forkortelsen brukes i forbindelse med RGB-kamera som beskrives i kapittel 2.3
- RGB-D - RGB - Dybde. En annen type RGB-kamera, der D-en står for dybde. Beskrives i kapittel 2.3
- AI - Artificial Intelligence/kunstig intelligens

Kapittel 2: Teori

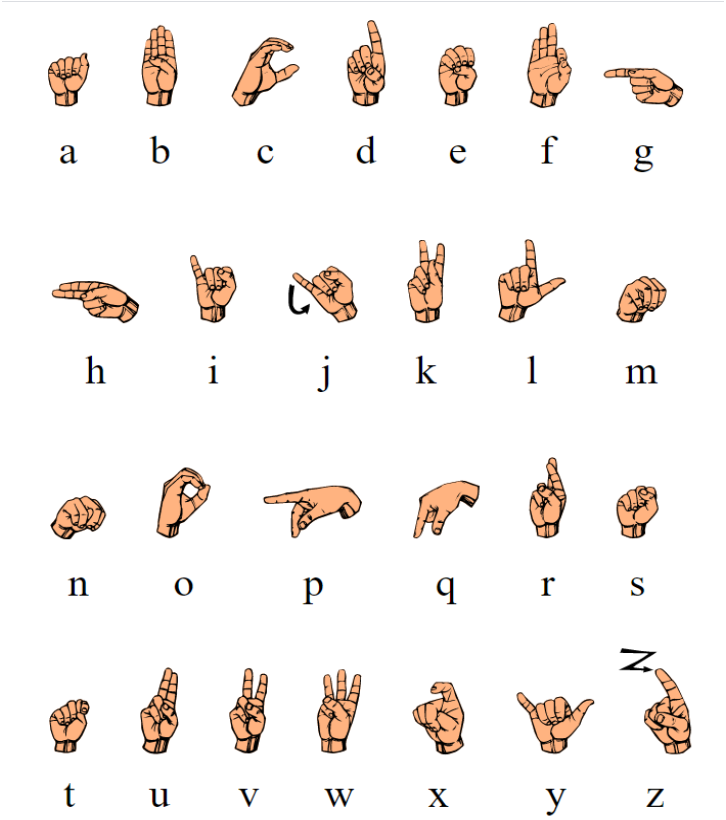
2.1 Tegnspråk

Tegnspråk er språk som bruker det visuelle for å kommunisere. Det er fullverdige, naturlige språk med egen grammatikk og leksikon. Med naturlig språk menes et språk som brukes av mennesker. Det finnes ikke noe universelt tegnspråk og det er ikke sikkert at mennesker som bruker forskjellige tegnspråk kan forstå hverandre, men det er også mange likheter mellom språkene. Tegnspråk brukes brukes primært av døve og stumme, men brukes også av de som har problemer med talespråk på grunn av en funksjonshemning eller tilstand og de med døve familiemedlemmer. Det er uvisst hvor mange tegnspråk som eksisterer i verden, men det er veldig vanlig at hvert land har sitt eget tegnspråk.

Både deler av kroppen samt ansiktet er viktig for tolking av tegnspråk. Håndflaten, fingerrotasjon, fingerposisjon, øyenbryn og munn er kroppsdeler som spiller en stor rolle. For eksempel kan én enkel ansiktsgrimase avgjøre forskjellen mellom to ord. Tegnspråk har som regel en egen kompleks grammatikk som er ulik talespråk.[3] Setningsoppbygging på Norsk tegnspråk bygges på måten: “tid - sted - jeg”. Så setningen “Jeg var på skolen i dag”, blir “i dag skole jeg”.[4]

2.1.1 ASL

ASL, eller American Sign Language, er et komplett, naturlig språk som har de samme språklige trekkene som talespråk. ASL benytter håndbevegelser og ansiktsuttrykk for å uttrykke seg, og er språket som brukes av døve og stumme i USA og store deler av Canada.[5]



Figur 2.1: De forskjellige bokstavene i ASL

Bildet er hentet fra wikipedia.[6]

2.2 Maskinlæring

Maskinlæring er en anvendelse av kunstig intelligens (AI) som går ut på at systemer, ved hjelp av algoritmer, er programmert til å automatisk lære og forbedre seg basert på erfaring, uten å være eksplisitt programmert til det. Maskinlæringsalgoritmer bruker treningsdata for å lage en matematisk modell for å kunne gjøre valg basert på mønstre i dataene. Etter at et system er ferdig opplært skal det gi en prediksjon basert på input-data det mottar. Maskinlæring brukes blant annet til talegjenkjenning, bildegjenkjenning og forutsi sekvenser. [7]

2.2.1 Nevrale Nettverk

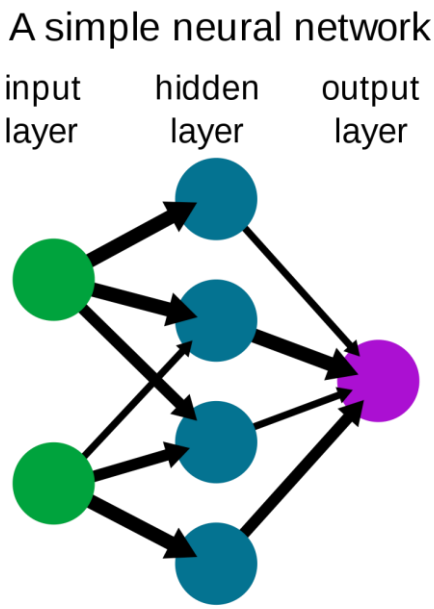
Kort sagt er et nevralt nettverk et nettverk av nevroner. Et nevralt nettverk kan enten være biologisk eller kunstig. Et biologisk nevralt nettverk tilsvarer nervesystemet som finnes i menneskekroppen, mens et kunstig nevralt nettverk (Artificial Neural Network) er et nettverk som består av kunstige nevroner brukt i maskinlæring.

Det finnes flere ulike kunstige nevrale nettverk innen maskinlæring. I dette prosjektet ble tre ulike testet, vanlig Artificial Neural Network (ANN), Recurrent Neural Network (RNN) og Convolution Neural Network (CNN).

2.2.2 Nettverkstyper

ANN

Et kunstig nevralt nettverk er en serie av algoritmer som ser etter underliggende sammenhenger i et sett med data ved hjelp av en prosess som etterligner måten en menneskelig hjerne fungerer. Akkurat som en menneskehjerne, består også et kunstig nevralt nettverk av nevroner, men i dette nettverk er nevronene matematiske funksjoner som samler og klassifiserer informasjon basert på valgt nettverksarkitektur. Det ligner mye på statistiske metoder som for eksempel regresjonsanalyse og kurvetilpasning. Nevronene, også kalt noder, er fordelt i lag, der alle nodene er knyttet sammen som vist i figuren til høyre. En maskinlæringsmodell kan bestå av ett eller flere lag. Et nettverkslag er et sett av noder som vanligvis mottar input fra et annet lag og sender den videre. Et lag er som regel uniformt, som vil si at det bare inneholder én type aktiveringsfunksjon. Det første og det siste laget i et nettverk kalles henholdsvis for input layer og output layer, mens alle lagene mellom kalles for "hidden layers" eller skjulte lag. De skjulte lagene knytter input-laget sammen med output-laget og transformerer input-data til output-data.



Figur 2.2: Simpelt nevralt nettverk

Forbindelsen mellom nodene kalles vekter. Vektene justerer input-verdiene når de sendes gjennom nettverket. På hver node er det en aktiveringsfunksjon som bestemmer signifikansen til nodens output når den sendes videre til neste node.[8] [9] Aktiveringsfunksjoner beskrives i 2.2.4.

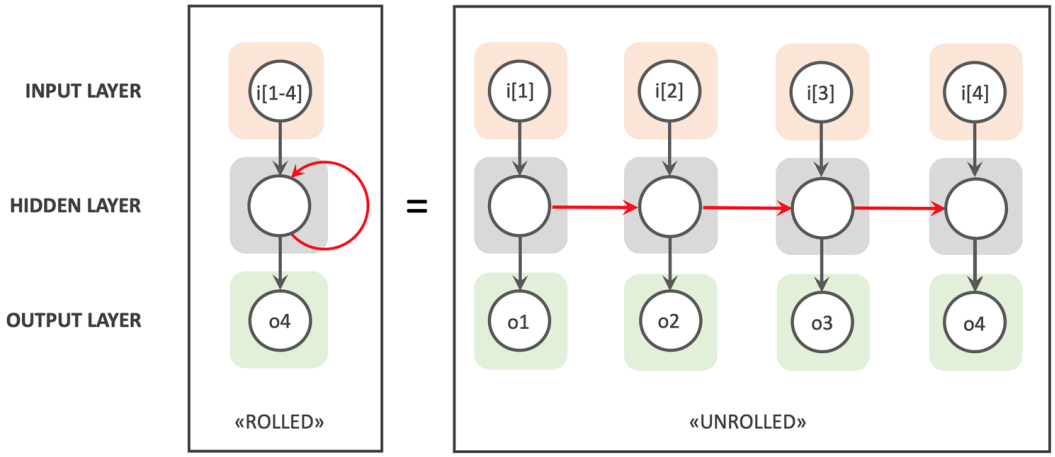
Dense

Et Dense layer er et vanlig lag med nevroner. Hver nevron mottar input fra alle de forrige nevronene i det forrige laget. Laget inneholder en vektmatrise, en biasvektor og alle verdiene fra forrige lag. Neste output blir regnet ut på følgende måte:$outputs = activation(inputs * kernel + bias)$
Der *activation* er hvilken aktiveringsfunksjon som brukes, *kernel* er vektmatrisen, *bias* er er verdi brukt i maskinlæring for å optimalisere modellen og input er dataen fra det forrige laget. Dense layer er det mest brukte laget og kan for eksempel brukes for å bestemme hvor mange outputs modellen skal ha. [10]

RNN

Et recurrent neural network er en type kunstig nevrale nettverk som er god til å trene på sekvensiell data. Nettverkstypen er mye brukt i tale- og tekstgjenkjenning. For at nettverket skal kunne se på en sekvens av data, legges det til en løkke. Løkken gjør at nettverket kan jobbe

seg gjennom den sekvensielle dataen samtidig som at tilstanden til de skjulte lagene beholdes mellom stegene.



Figur 2.3: Modell av et RNN

Bildet over viser et eksempel på hvordan et recurrent neural network fungerer. På venstre side av likhetstegnet ser man et simpelt RNN med input, hidden og output noder. Nettverket tar inn sekvensiell data med lengde fire, looper gjennom denne og gir o4 som output. For hver iterasjon arver hidden layeret et slags arbeidsminne fra de forrige iterasjonene, symbolisert av den røde pilen. Etter at data i hver node har blitt sendt gjennom en aktiveringsfunksjon, sendes den videre til både output-laget og til neste iterasjon av nettverket. Høyre siden av likhetstegnet viser iterasjon for iterasjon i en “utrullet” modell. Et RNN kan ses på som en rekke av flere ANN der hver ANN er en kopi av RNN-et med de samme vektene og aktiveringsfunksjonene. [11]

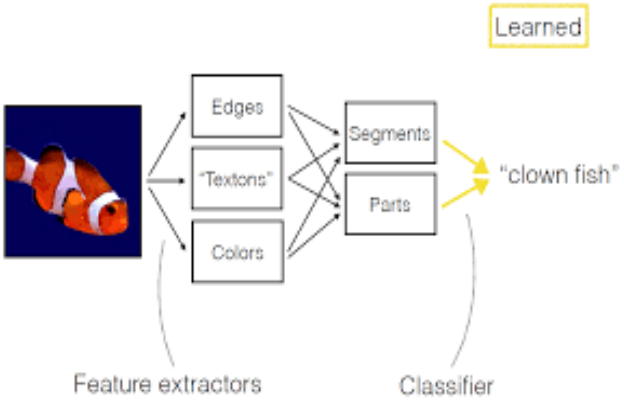
LSTM

LSTM eller Long short-term memory er en spesialisert type recurrent neural network brukt til å modellere sekvensiell data og blir ofte brukt i språkprosessering. Fordelene med et LSTM over RNN er at LSTM tar vare på viktig informasjon som ble lært i tidligere iterasjoner, over en lengre periode. Dette fører til at denne informasjonen har større innflytelse over modellens valg i senere iterasjoner. En LSTM-enhet består blant annet av en “forget gate” som bruker en aktiveringsfunksjon for å avgjøre om informasjonen skal lagres eller ikke. [12]

En annen type RNN er GRU (Gated Recurrent Unit). GRU er en porteringsmekanisme for å kontrollere flyten av informasjon mellom noder i nettverket. Fordelen med GRU er at det effektivt klarer å opprettholde langsiktige avhengigheter i sekvensiell data. Grunnen til dette er fordi GRU har færre treningsparametere og bruker derfor mindre minne. Av den grunn fungerer GRU best på mindre datasett med kortere sekvenser mens LSTM fungerer best på større datasett med lengre sekvenser. [13]

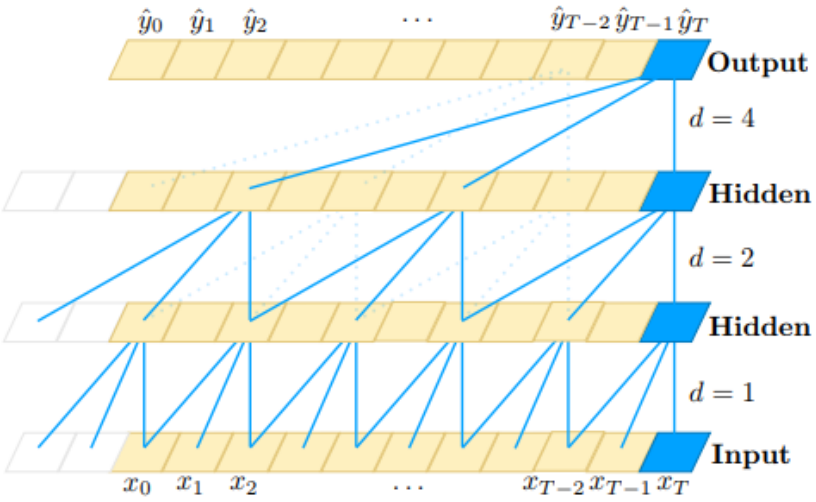
CNN

Convolutional neural network eller CNN er en type nevralt nettverk som er mest brukt til å analysere bilder, men brukes også innen språkgjenkjenning. CNN har hidden layers som kalles convolutional layers, og i disse lagene er det filter som ser etter mønstre. I de første lagene kan filtrene se etter enkle mønstre som firkanter eller linjer, men i noen av de dypere lagene kan de se etter ting som øyner eller sko.[14]



Figur 2.4: Filtrering i CNN

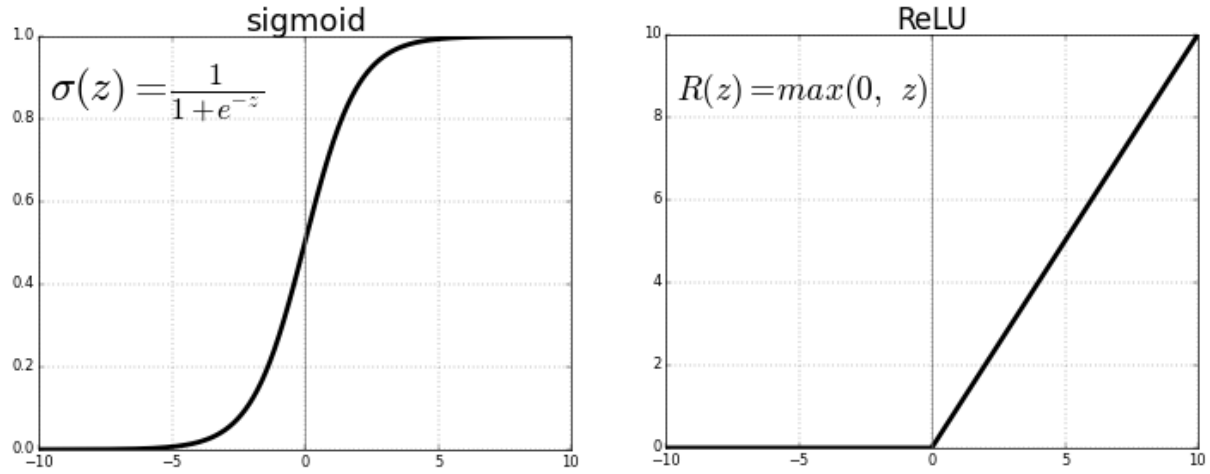
Temporal Convolutional Network eller TCN er en type CNN som egner seg bra til sekvensiell data. TCN ser direkte på tidligere input og bruker disse for å bestemme output. TCN har også muligheten til å være non-causal som vil si at den tar fremtidige inputs inn i beregningen. Dette kan være bra hvis det skal oversette en setning der modellen kan se på framtidige ord. TCN har på mange områder overgått RNN som vist i denne rapporten. [15] [16]



Figur 2.5: Virkemåten til TCN

2.2.3 Aktiveringsfunksjoner

En aktiveringsfunksjon er en funksjon som kjøres på output-verdiene på hver node i hvert lag og sendes deretter videre til neste lag. Funksjonene kan være både lineære og ikke-lineære og kartlegger verdiene mellom to grenseverdier basert på aktiveringsfunksjonen. Ikke-lineære funksjoner er de mest vanlige og figuren under viser to eksempler på denne type funksjon.[17]

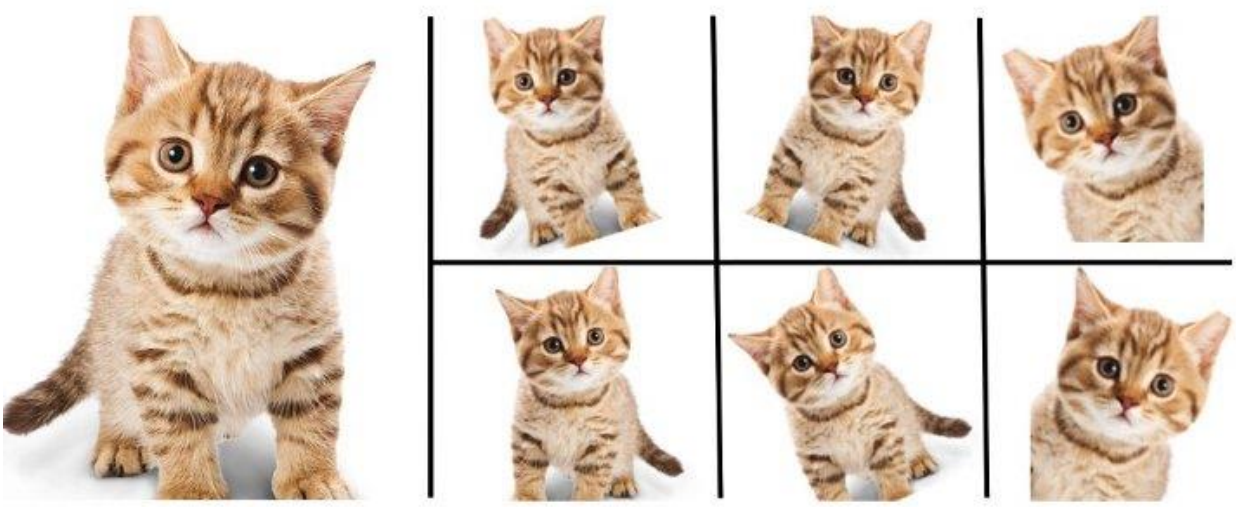


Figur 2.6: Funksjonen til Sigmoid og ReLu

Tilhørende hver type aktiveringsfunksjon er det vanlig å bruke en kernel initializer. En kernel initializer initialiserer vektene mellom hver node til en verdi som er hensiktsmessig å bruke for aktiveringsfunksjonen.

2.2.4 Data augmentation

Data augmentation eller datautvidelse vil si å øke mengden data ved hjelp av ulike metoder. Datamengde kan ofte gjøre maskinlæringsmodeller mer treffsikre og bedre generalisert. Data augmentation bruker dataen som allerede er samlet, til å øke datamengden. Det er mange teknikker for å øke datamengden på, og noen teknikker kan bli brukt om hverandre. Dette gjør slik at datamengden kan mangedobles på kort tid.

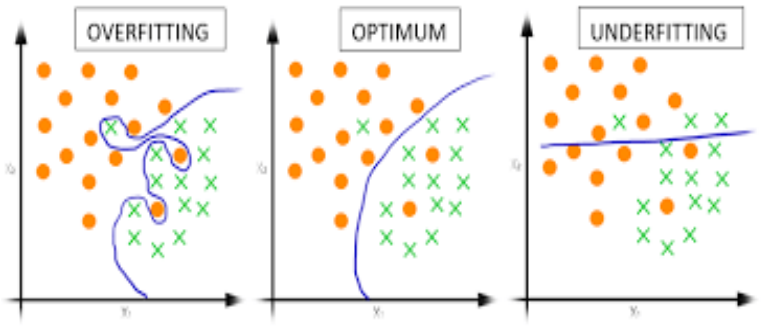


Figur 2.7: Eksempel på data augmentation

Data augmentation brukes på mange områder innen maskinlæring. Den vanligste vil være å bruke data augmentation på bilder. Bildet over viser noen teknikker for å øke mengden data. Teknikkene som vises er blant annet rotering, skalering og klipping. Data augmentation-teknikker kan i noen tilfeller virke på områder de ikke var designet for med mindre forandringer. [18]

2.2.5 Overfitting og underfitting

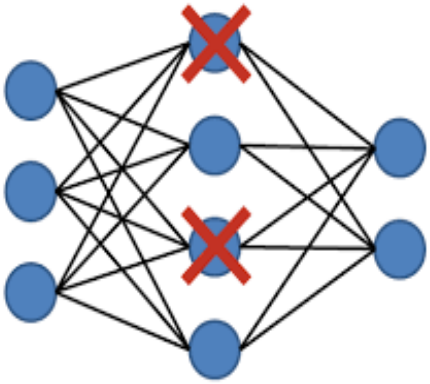
En måte å vurdere hvor god en maskinlæringsmodell er, kan være å se på hvor bra den klassifiserer generell data. For å se hvor dårlig en modell er, kan begrepene underfitting og overfitting benyttes. Overfitting betyr at modellen lærer seg treningsdataen for bra, mens underfitting betyr at en modell ikke klarer å generalisere bra på treningsdata eller ny data.



Figur 2.8: Modellering av overfitting og underfitting

Overfitting skjer når en modell lærer seg detaljene og støyet til datasettet i så stor grad at det har negativ påvirkning på modellen. Det betyr at støyet i dataen blir plukket opp som en egenskap av dataen. Det gjør slik at modellen ikke klarer å generalisere bra på annen data fordi denne dataen ikke inneholder samme type støy. Overfitting kan oppstå hvis man for eksempel har ujevn data eller for stor modell. Størrelsen på modellen vil påvirke hvor mange av egenskapene til dataen modellen klarer å lære. Hvis den er for stor vil den lære støyen. Mindre modeller vil ta ut nøkkelegenskaper, men for liten modell vil ikke finne nok egenskaper til å generalisere bra og vil bli underfitted.[19]

Dropout er en teknikk for å minske overfitting. Denne teknikken handler om å sette en brøkdel av input-verdiene til null på hver oppdatering når den trener og brukes i mellom hvert lag.[20]



Figur 2.9: Viser dropout i effekt

2.3 Bevegelsesfangst

Tegnspråk handler om å bevege på hendene og andre kroppsdeler for å bli forstått. Man trenger derfor en måte å fange opp bevegelsene på for å kunne forstå hva de mener. Teknologiene som finnes bruker forskjellig maskinvare, og dette er viktig å tenke på ettersom ikke alle har tilgang til de samme ressursene. De forskjellige teknologiene gir også forskjellig kvalitet på dataen de genererer. Noen er veldig presise mens andre ikke har samme nøyaktighet.

RGB Kamera

RGB kamera er det vanligste kameraet. Kamera fanger opp lys i forskjellige bølgelengder for å lage et bilde. For å fange opp bevegelser med et kamera må man ta video. For en tegnspråkstolk vil dette være bra nok for å kunne oversette hva personen i videoen formidler. Det viser seg at det er en del vanskeligere å forstå for en datamaskin.

RGB-D kamera

RGB-D kamera er et vanlig RGB kamera som har en ekstra sensor. Denne sensoren måler dybden til bilde og lager et dybdekart. Dybde-kartet og det vanlige RGB bildet kombineres for å få økt mengde data per pixel. Det vil si at hver pixel i et bilde eller frame i en video har en dybdeverdi. Dybdekamera bruker en infrarødt prosjektor til å regne ut dybden og posisjonen til punktene. En av svakhetene til dybdekamera er at sollys og andre dybdekamera vil lage forstyrrelser for dybdesensoren. [21]

Stereokamera

Stereokamera har to eller flere sensorer som tar opp video eller bilder samtidig. Dette gjør det mulig å simulere menneskelig syn og derfor også en tredimensjonal modell av bildet. Ved hjelp av to eller flere sensorer kan posisjonen til et punkt i bildet kalkuleres. Dette gjøres med å finne krysningspunktet mellom to linjer fra to sensorer. Sluttproduktet av dette gir en tredimensjonal modell av et bilde eller en video. [22]

Markørbasert sporing

I markørbasert sporing brukes det markører som er plassert på en person eller et objekt som skal spores. Det brukes flere kameraer til å spore markørene. Markørene blir plassert på ulike deler av kroppen for å spore bevegelsene. Hver markør vil ha en verdi i et xyz-koordinatensystem for hvert tidsinterval. Dette er en effektiv og presis måte å lage animasjoner på.

Kapittel 3: Valg av teknologi og metode

For å gjenkjenne tegnspråk kreves det nøyaktige og pålitelige teknologier. Teknologiene må blant annet kunne fange opp statiske og dynamiske bevegelser, videre må dataoppfangsten fra bevegelsene kunne brukes i en maskinlæringsalgoritme for å gjenkjenne hvilken bevegelse som ble utført. Det vil derfor kreve at flere teknologier jobber sammen for å oppnå et bra resultat, men mange av teknologiene har både fordeler og ulemper. Da det ikke finnes mye forskning på hva som er nødvendig for bevegelsesfangst, ble det prioritert teknologier som er lett å bruke og ikke krever mye utstyr. Videre i kapittelet diskuteres også fordeler og ulemper med de forskjellige teknologiene både for bevegelsesfangst og maskinlæring.

3.1 Drøfting av teknologiene

Det finnes mange relevante teknologier som kan brukes i dette prosjektet. For bevegelsesfangst er Kinect eller Leap Motion gode alternativer, mens andre muligheter er deteksjonsbibliotekene Openpose og Mediapipe. Når det kommer til databehandling og maskinlæring er blant annet pandas mye brukt til databehandling og tensorflow og keras mye brukt til maskinlæring. Andre eksempler på biblioteker som kan brukes er Dlib og OpenCV. Dlib er et C++-basert bibliotek for maskinlæring og databehandling, mens OpenCV er et rammeverk brukt til full kroppsdeteksjon i python. [23] [24]

Mediapipe og Openpose er to teknologier som kan estimere leddposisjoner og ansiktsuttrykk fra video. Dette passer bra ettersom tegnspråk kan ha flere ord som har lik håndbevegelse, men har forskjellig betydning ettersom hvordan ansiktsuttrykk som blir vist. Mediapipe kan bli mer skreddersydd til egendefinert bruk, derimot er Openpose enklere å bruke, men ikke like lett å gjøre endringer på. Openpose og Mediapipe har to hovedforskjeller som har stor betydning for deres effektivitet i prosjektet. Prosessorkraft og nøyaktighet er de to hovedforskjellene, der Openpose er mer nøyaktig, men bruker mer prosessorkraft. Openpose kan bruke video fra stereokamera som vil gi sporingspunkter i XYZ-rommet, men med vanlig RGB-video vil den i likhet med Mediapipe gi sporingpunkter i XY-rommet.[25] [26]

Manomotion og Uens er rammeverk for å fange opp posisjonen til en hånd i sanntid. Disse teknologiene klarer det med minimal mengde prosessorkraft, og av den grunn kan det brukes i en mobiltelefon. Fordelene med disse teknologiene er at de kan brukes i en vanlig smarttelefon og vil derfor kunne være en mulighet for utvikling av en app. Ulempen er at disse bare kan brukes til å fange håndbevegelser, og vil derfor kun bli brukt til tegnspråks staving.[27] [28]

Markørbasert sporing blir brukt på mange områder innenfor blant annet filmindustri, robotikk og spillutvikling. Personen som spores må vanligvis ha markører på kroppen. Posisjonen til markørene blir fanget opp av et sett med kameraer som gir posisjonen i XYZ-rommet. I nyere tid er det mulig å droppe markørene for noe mindre nøyaktighet. Grunnen til at teknologien er mye brukt er at den er veldig nøyaktig. Ulempen med et slikt system er at det kreves del utstyr, og blant annet et område som blir brukt til dette.[29] [30]

Tensorflow er en open source plattform for maskinlæring, som inneholder ressurser og bibliotek som lar brukere enkelt utvikle maskinlæringsbaserte applikasjoner. Keras er et høynivå API, skrevet i Python som gjør det enklere å implementere nevrale nettverk i applikasjoner. Sammenhengen mellom de to er at Keras er basert på tensorflow. Fordelen med å bruke nevnte teknologier mot ikke å bruke de, er at man sparer mye tid og ressurser. Man trenger ikke å bruke tid på å programmere nevrale nettverk, lag, aktiveringsfunksjoner og loss-funksjoner fra bunnen av. Istedenfor kan man bruke tid på å lage flere forskjellige modeller for å sammenligne de og finne ut av hvilke modeller som presterer best til ønsket formål eller oppgave. [31] [32]

I tillegg til teknologiene nevnt ovenfor ble det også undersøkt flere teknologier som løser samme problem. Det er også flere ulike metoder å løse prosjektoppgaven på med samme teknologier, men valg av teknologi og metodologi begrunnes videre i rapporten og baseres mye på preferanse, tidligere erfaringer og simplisitet.

3.2 Valg av teknologi

Det første som ble prioritert når det kom til valg av teknologi var å finne ut hva som var relevant for tolkning av tegnspråk. For å finne ut av dette ble det blant annet foretatt et intervju med en tegnspråkstolk og det ble gjort undersøkelser på nett. Dette påvirket hvilke teknologier som ble tatt i betraktning. Disse teknologiene måtte kunne spore hender, armer, overkropp og ansiktsuttrykk. Hovedfokuset ved valg av teknologi var også at det skulle være enkelt å bruke, men det enkle har også vanligvis ulemper som i dette tilfelle er nøyaktigheten til sporingen. Noe som gjør det enkelt, er å bruke vanlig video der vi ikke har dybde. Dette gjør at dataen vi får ut av videoene er i XY-rommet istedenfor XYZ-rommet. Da mister man en tredjedel av informasjonen, noe som kan påvirke resultatene. En annen faktor som påvirket valgene var erfaring og tidligere kunnskap.

Basert på prioriteringene ble Openpose og Mediapipe valgt for sporing, da begge disse bare trengte vanlig video for å spore bevegelser der andre trengte spesialutstyr. Dette passet også bra fordi datasettet måtte lages fra bunnen av, og videoer fra andre kilder kan da brukes som testdata. Begge deteksjonsbibliotekene var også interessante da Openpose ble nevnt av prosjektveilederen og Mediapipe er utviklet av Google. Dette ga høyere forventninger for disse deteksjonsbiblioteker enn andre. Openpose og Mediapipe er også open source, noe som gjorde det enkelt å installere og de hadde aktive brukere som diskuterte problemer. Testing av disse to kan være relevant da det vil finne ut om 2D-sporing vil gi tilfredsstillende resultater.

For å lage maskinlæringsmodeller ble Tensorflow og Keras valgt som bibliotek. Disse ble valgt fordi de var kjent fra før av og er veldig mye brukt i maskinlæringssamfunnet. Det finnes derfor mye dokumentasjon og eksempler på både installasjon og brukt av begge bibliotekene. Tensorflow kan også kjøres på Nvidia-skjermkort, noe som gjør det mye raskere å utvikle modellene. En annen grunn til at bibliotekene ble valgt var at prosjektet ble mer rettet mot å teste ulike maskinlæringsmodeller, og dette er spesielt enkelt å få til med Keras.

3.3 Metode og framgangsmåte

I dette kapittelet skal det redegjøres for oppgavens metode og framgangsmåte. Valgene som ble tatt i dette prosjektet vil bli beskrevet i underkapitlene og vil vise hvordan oppgaven ble gjennomført og hva tankegangen bak valgene var.

3.3.1 Datainnsamling

Det første som ble undersøkt når det kom til datainnsamling var om det eksisterte en database med tegnspråksdata som kunne brukes. Da det er ganske vanskelig å lage god konsistent tegnspråksdata, var det ikke lett å finne noe som passet prosjektet. Det beste som ble funnet var en amerikansk database som verken var ferdiglaget eller åpen for allmenn bruk. Det ble da vedtatt å lage et eget datasett. Datasettet ble laget ved å ta opp flere korte videosnutter av bokstavene fra ASL ved hjelp av et RGB-kamera. Videosnuttene måtte inkludere både albuen og den ene skulderen da Openpose trengte dette for å kunne klare å spore en hånd. Måten datasettet ble gjort konsistent på var å lage like lange videosnutter for hver bokstav, foreta samme mengde bevegelse i hver videosnutt, holde hånda i cirka like posisjoner og å alltid bruke samme hånd.

Neste steg i datainnsamlingsprosessen var å prosessere videoene med Openpose og Mediapipe. Openpose hadde allerede en metode for å generere en JSON-fil per frame som inneholdt x-koordinat, y-koordinat og en nøyaktighetsverdi for hvert av de 21 punktene på hånda. Det ble derfor bestemt at noe lignende måtte gjøres med Mediapipe. For å få dette til måtte kildekoden til noen av Mediapipe-filene redigeres. Det eksisterte allerede et objekt som inneholdt metoder for å hente ut både x- og y-koordinater. Da gjenstod det bare å skrive kode for å skrive x- og y-koordinatene til en JSON-fil cirka likt strukturert som JSON-filene generert av Openpose.

For å forenkle prosessen med å måtte kjøre Openpose- og Mediapipe-sporingen på hver enkelt video, ble det laget skriptfiler som søkte gjennom bestemte mapper og utførte denne operasjonen på alle videofilene liggende i mappen. Basert på hvilken ASL-bokstav som ble vist i videoen, fordelte skriptet de genererte JSON-filene utover mapper med navn tilsvarende bokstav vist i videoen. På denne måten ble det lettere å knytte JSON-filene opp mot tilhørende bokstav, noe som gjorde databehandlingsprosessen enklere.

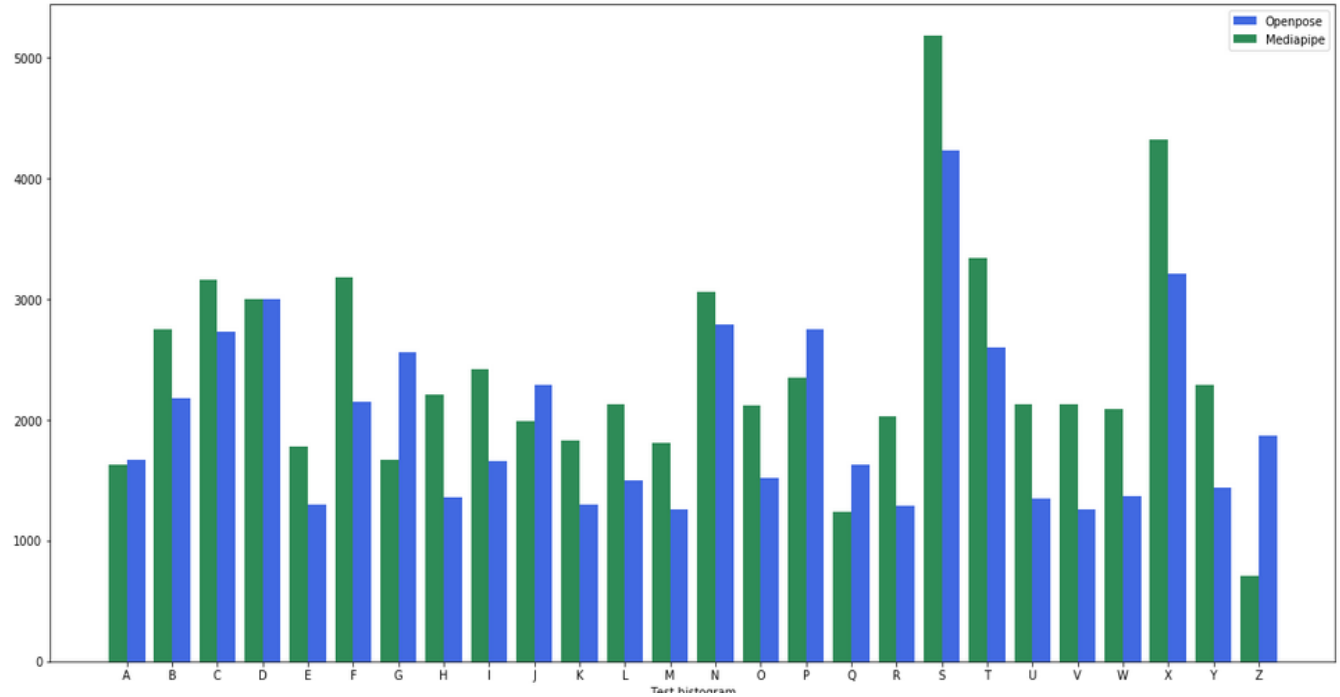
3.3.2 Behandling av data

Det første som måtte gjøres med dataen var å lese den inn i arrays. Til det ble det valgt å bruke jupyter notebook sammen med biblioteket pandas, som er et databehandlingsbibliotek for python.[33] Det ble derfor skrevet kode som leste inn JSON-filene, hentet ut x- og y-koordinatene samt sannsynlighetsverdiene fra Openpose og strukturerte dataene i en Dataframe som vist i figuren under. Samtidig som hver verdi ble lest inn ble to normaliseringsmetoder implementert. Den ene metoden som ble implementert var å alltid translatere punktene slik at første punktet i hver frame ble satt i (0,0). Den andre metoden gikk ut på å lese inn z-skåren til hver verdi med hensyn på verdiene fra samme frame. Z-skåren ble regnet ut for x og y hver for seg og ble regnet ut på følgende måte: $z = (x - \mu)/\sigma$ der μtilsvarer gjennomsnittet og σtilsvarer standardavviket. Sammen med hver rad av keypoints ble også en kolonne med verdier fra 0 til 25 lagt til for å representere en av de 26 bokstavene i ASL-alfabetet. Sannsynlighetsverdiene ble senere fjernet til fordel for bedre nøyaktighet.

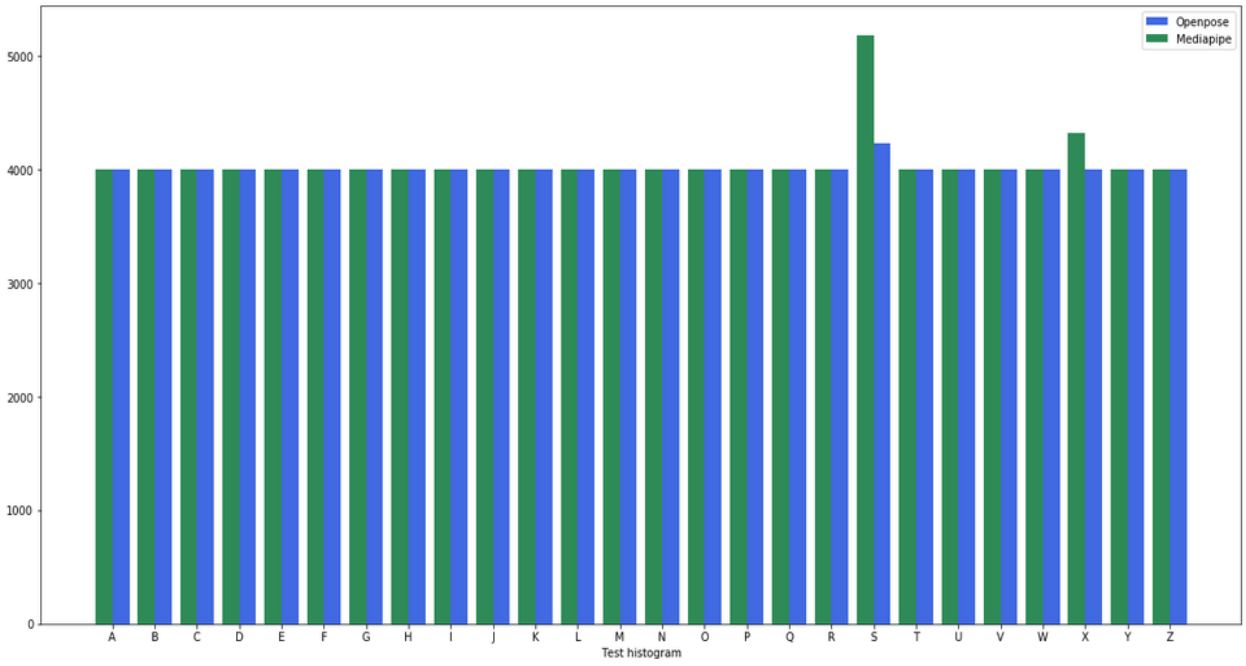
	Letter	keypoint_0_x	keypoint_0_y	keypoint_1_x	keypoint_1_y	keypoint_2_x	keypoint_2_y	keypoint_3_x	keypoint_3_y	keypoint_4_x	...	keypoint_16_x
0	0	0.0	0.0	33.923	-22.616	73.500	-56.539	100.356	-98.943	134.279	...	1.413
1	0	0.0	0.0	35.815	-23.418	75.763	-60.611	100.559	-100.559	137.752	...	6.887
2	0	0.0	0.0	35.986	-22.145	74.740	-56.747	99.653	-96.885	135.639	...	6.921
3	0	0.0	0.0	33.591	-20.995	71.382	-55.986	96.576	-96.576	134.367	...	2.799
4	0	0.0	0.0	34.991	-19.595	75.580	-54.585	99.374	-95.175	134.365	...	5.598

Figur 3.1: Utsnitt av dataene i Dataframe

Data augmentation var også noe som ble testet på treningsdataen. Måten dette ble gjort på var ved å rotere hvert punkt rundt origo med en tilfeldig vinkel. Alle punktene for hver frame ble naturligvis rotert med samme tilfeldige vinkel. Som man kan se av figur 3.2 og figur 3.3 har treningsdatasettet blitt større og jevnere fordelt.

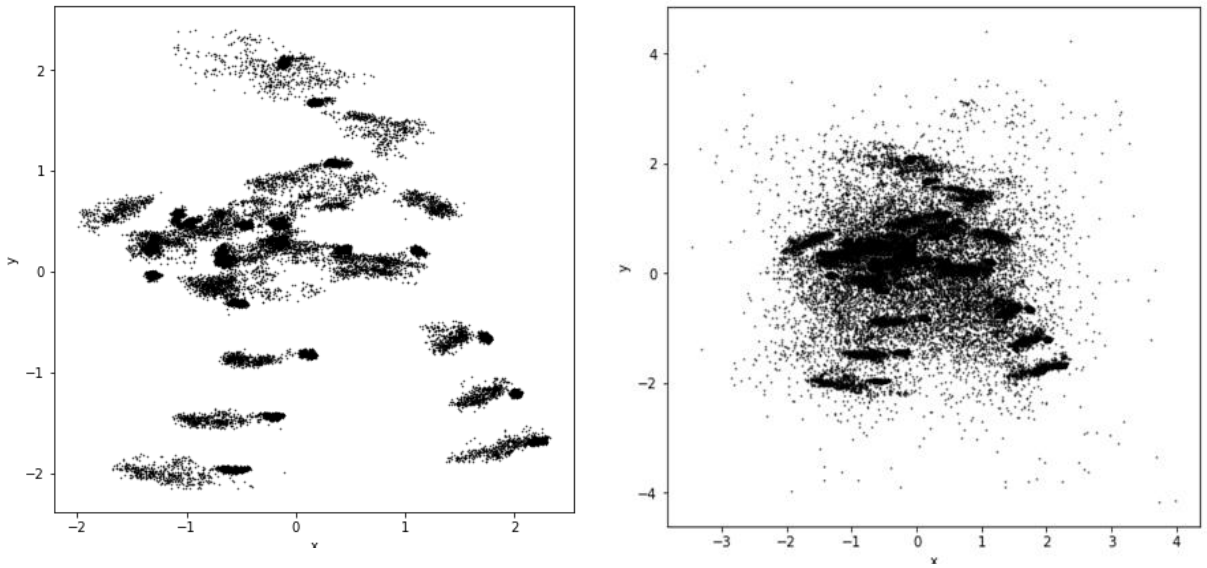


Figur 3.2: Fordelingen av data.



Figur 3.3: Fordeling av data etter data augmentation

Fjerning av avvik i dataen er også en metode som kan forbedre modellen. Dette ble utført ved å ta for seg hver kolonne for hver bokstav og kalkulere z-skåren for hver verdi i kolonnen relativt til kolonnens gjennomsnitt og standardavvik. Videre bruker man absoluttverdien til z-skåren og velger om man vil beholde eller fjerne raden basert på en grenseverdi. Grenseverdien ble i dette tilfelle valgt til å være 4.



Figur 3.4: Punktplotting for èn bokstav etter(venstre) og før(høyre) fjerning av avvik.

3.3.3 Maskinlæringmodellene

Som det fremgår av oppgaveteksten ble prosjektet rettet mot å teste ulike maskinlæringsmodeller og sporingsteknologiene Mediapipe og Openpose for å finne det beste alternativet for tegnspråksgjenkjenning. Dette kapittelet inneholder en detaljert beskrivelse av hvordan de ulike nettverkene ble bygget opp med forskjellige lag, aktiveringsfunksjoner og dropout. Alle maskinlæringsmodellene som ble implementert fungerer på forskjellige måter og har alle fordeler og ulemper. For eksempel vil en RNN-modell kunne huske hva som hadde skjedd i tidligere iterasjoner og har da mer informasjon tilgjengelig for å predikere neste tegn. Tidligere har RNN-modeller blitt brukt til å gjenkjenne tale[34], noe som var mye av grunnen til at denne typen modell ble implementert. Det ble også testet CNN-, TCN- og ANN-modeller. Fordelen med CNN er at den er god til å se etter mønstre, men en ulempe er at det vanligvis brukes i forbindelse med bildegjenkjenning eller lignende. Det finnes derfor lite eksempler og informasjon om bruk av CNN til andre tilfeller. TCN er en type CNN som har muligheten til å se direkte på tidligere input. TCN har også fått bedre resultater enn noen av de beste RNN-resultatene[35], noe som gir grunn til å teste modelltypen. Noen av utfordringene er at modellene må kunne skille mellom veldig like statiske tegn og kunne spore bevegelser, noe som gir

modellene som kan “huske” de tidligere data en fordel. Det finnes heller ikke noen fasit for hvordan lagene i modellene skal bygges opp, så det måtte testes for å se hva som ga best resultat på datatypen. Resultatene under testing ble brukt til å finne den beste lagoppbygningen for de forskjellige modellene.

ANN modellen ble bygget for å teste hvor bra en enkel modell kunne gjenkjenne tegnspråk. Ettersom dataene fra Mediapipe og Openpose var forskjellige måtte disse testes hver for seg med hver sin modell. Mediapipe ble droppet i senere modeller fordi den ga ikke tilfredsstillende resultater i forhold til Openpose. Modellen er bygget opp av flere dense-lag med ulik størrelse. Input-laget består av 672 nevroner og bruker kernel initializeren som heter “glorot_uniform”. Størrelsen på skjulte lagene halveres for hvert lag fra 336 til 21 nevroner. Hvert av de skjulte lagene samt input-laget bruker aktiveringsfunksjon ReLu. Output-laget har 26 nevroner og bruker aktiveringsfunksjonen “softmax”. Dette gir en output-vektor som inneholder 26 prosent-verdier der hver verdi beskriver sjansen for hvert tegn. Openpose hadde dropout 0.08 på innputlaget og 0.04 på de skjulte lagene, mens Mediapipe hadde 0.5 på input-laget og 0.25 på de skjulte lagene.

Da RNN-modellen ble bygd, ble flere ulike RNN-lag testet. I starten ble forskjellige kombinasjoner av LSTM- og GRU-lag implementert med varierende størrelse. Da dette ikke ga resultater tilsvarende ANN-modellen, ble til slutt et SimpleRNN-lag brukt. Laget er bygd opp av 150 nevroner og fungerer som input-laget i modellen. Output-laget er likt som i ANN-modellen og inneholder 26 nevroner og bruker softmax som aktiveringsfunksjon.

CNN modellen er bygd opp med et convolutional-lag med 128 filtre, kernelstørrelse på to og brukes som input-lag. Deretter brukes det et Max-Pooling-lag som kartlegger egenskaper i dataen. De to neste lagene er to dense-lag, begge med aktiveringsfunksjon ReLu, der første lag har 256 nevroner og andre lag har 64. Output-laget har 26 nevroner og aktiveringsfunksjon “softmax”. Mellom hvert lag ble det implementert et dropout-lag med 0.1 dropout.

Forskjellen på RNN- og CNN-modellen i forhold til de andre, er at begge modellene bruker 10 frames som input-data istedenfor 1. Dette ble gjort for å få med mere av bevegelsen i de dynamiske ASL-tegnene J og Z, noe som kunne føre til bedre resultater. Dette påvirker hvor rask modellen er ettersom det avhenger av hvor mange bilder i sekundet kameraet tar opp. Tar kameraet 30 bilder i sekundet vil det si tre gjetninger i sekundet.

TCN modellen har ett convolutional-lag hvor det er 70 filtre. Deretter har laget 6 rest blokker som ser på tidligere input og blir sendt til output-laget som er et dense-lag med 26 nevroner og aktiveringsfunksjonen “softmax”.

Kapittel 4: Resultater

I dette kapittelet legges det fram resultater fra de forskjellige modellene fremstilt i tabeller og grafer samt forklaringer av dataene. Resultatene vil også bli beskrevet i forhold problemstillingen forklart i innledningen og til målene satt i starten av prosjektet. Først presenteres de beste resultatene oppnådd med både mediapipe og openpose og alle de fire ulike modellene. Videre forklares resultatene oppnådd med å teste ulike databehandlingsmetoder.

Resultatene ble produsert ved å trene maskinlæringsmodellene på det selvlagde datasettet med en læringsrate på 0,001, som endres under treningen. På ANN-modellen brukes en fast læringsrate på 0,0001. Læringsraten indikerer hvor mye vektene justeres av modellen under trening. Antall epoker brukt i de forskjellige modellene varierer da noen modeller bruker høyere batch size enn andre, men resultatene fra RNN-, CNN- og TCN-modellene kan oppnås ved å bruke cirka 50 epoker. Antall epoker sier noe om hvor mange ganger en modell skal kjøre gjennom treningsdataene, mens batch size sier noe om hvor stor del av dataene modellen skal trene på for hver iterasjon. Andre modellparametre brukt til å fremstille resultatene beskrives i kapittelet ovenfor. Resultatene indikerer hvor godt modellen klarer å gjenkjenne de forskjellige ASL-tegnene.

Tabellen under viser høyest oppnådd nøyaktighet til to ANN-modeller som bruker data generert fra samme videoer. Treningsdata og Testdata er en oppdeling av det originale selvlagde datasettet. Grunnen til denne oppdelingen er for å validere om modellen klarer å gjenkjenne data den ikke har trent på. Valideringsdata er data generert på samme måte men hentet fra en annen kilde. Dette er en bedre måte å validere om modellen gjenkjenner helt ulik data.

Modell:	Mediapipe ANN	Openpose ANN
Treningsdata	95,9%	99,9%
Testdata	96,8%	99,3%
Valideringsdata	68,9%	94,8%

Tabell 4.1

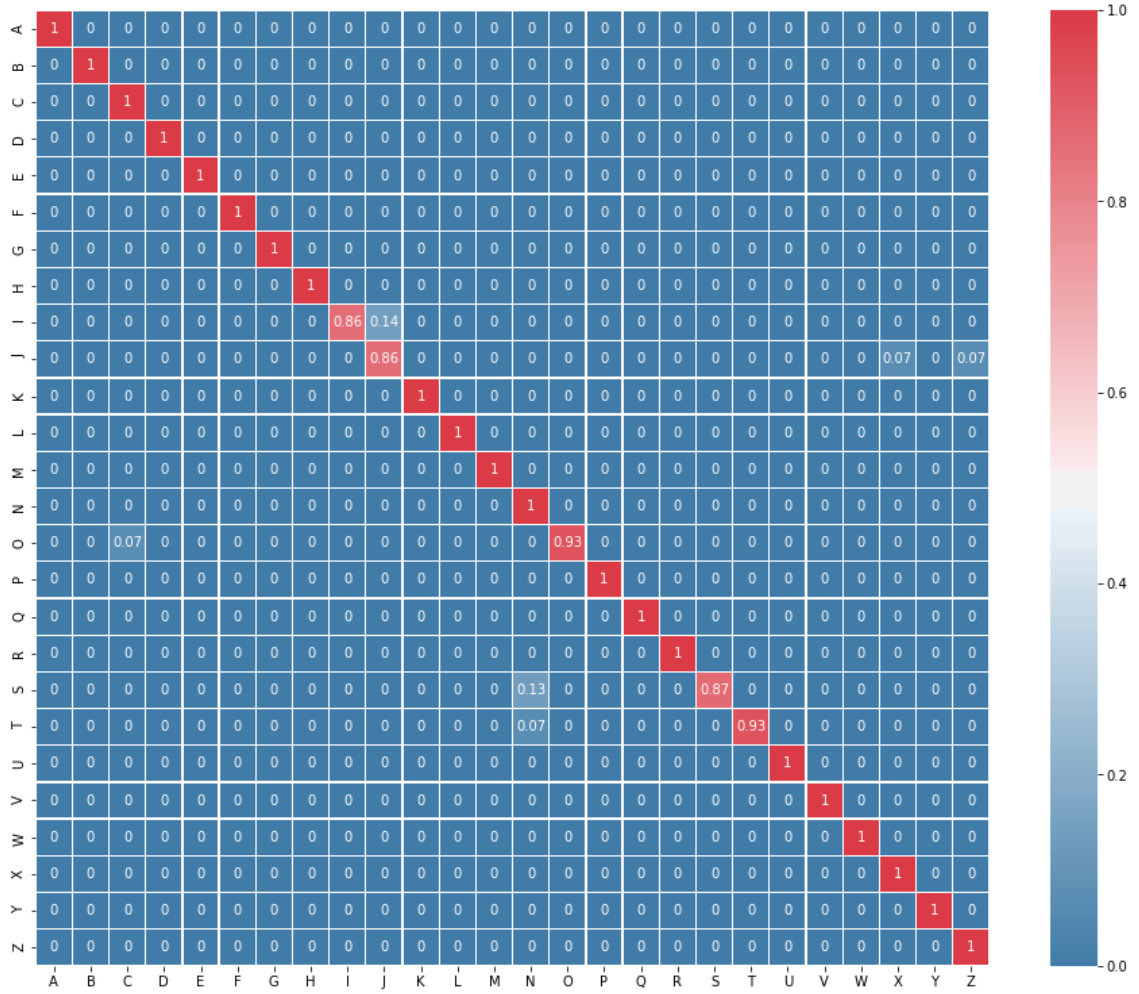
Resultatene viser klar forskjell mellom modellenes tolkning av valideringsdata. Man ser at Mediapie har klart lavere nøyaktighet på valideringsdataene, som vil si at modellen ikke er veldig flink til å generalisere seg, derimot har Openpose høy nøyaktighet på både testdata og valideringsdata.

Tabellen nedenfor viser sammenligningen av de fire ulike modellene som ble implementert der alle bruker Openpose. I dette tilfelle er også datasettet delt opp på samme måte og valideringsdataen er hentet fra samme kilde. Man kan se at alle modellene presterer cirka like bra på treningsdataen og testdataen med unntak av RNN-modellen som oppnår 5% lavere nøyaktighet på testdataen. På valideringsdataen oppnår modellene relativt like bra nøyaktighet der CNN-modellen yter best i forhold til de andre.

Modell:	ANN	RNN	CNN	TCN
Treningsdata	99,6%	97,4%	95,7%	97,6%
Testdata	99,5%	92,4%	96.1%	97,1%
Valideringsdata	94,8%	93,1%	97,8%	93,4%

Tabell 4.2

Heatmappet under viser hvordan CNN-modellen presterer på de ulike bokstavene i valideringsdataen. Vedlagt ligger lignende heatmap for de tre andre modellene(ref). Det man ser av de forskjellige heatmapene er at det er lavest nøyaktighet på de dynamiske ASL-tegnene **J** og **Z**. **J** er et dynamisk tegn som vises som **I** i starten av bevegelsen, dette kan derfor være grunnen til at **I** oversettes til **J** 14% av gangene. Forskjellene på de ulike heatmapene er at noen av modellene sliter med forskjellige bokstaver.



Figur 4.1: Heatmap over CNN-modellens gjetninger.

Resultatene under viser hvilken bruk av databehandlingsmetoder som ga best resultater. Da poenget med disse resultatene er å teste metodene ble modellene kjørt med bare 10 epoker, med unntak av ANN som ble kjørt med 100. Dette ga gode nok resultater til å kunne se fordelene og ulempene med metodebruken. I tabell 4.3 der fjerning av avvik mot ikke fjerning av avvik ble testet, ser man at de fleste modellene fikk bedre resultat med sistnevnte metode. Alle

modellene med unntak av ANN oppnådde cirka to prosent høyere nøyaktighet uten fjerning av avvik.

	ANN	RNN	CNN	TCN
Ikke fjerning av avvik	86,3	91,5	92,5	93
Fjerning av avvik	88,8	90,2	90,7	92,1

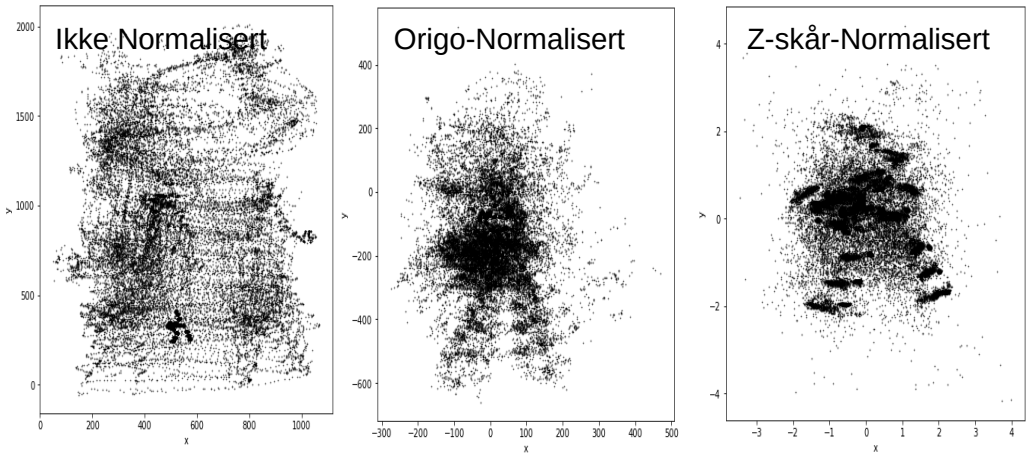
Tabell 4.3

Videre ble ingen normalisering, origo-normalisering og z-skår-normalisering testet på alle fire modellene. Med origo-normalisering menes alltid å translatere punktene slik at det første punktet i hver frame havner i origo. Z-skår-normalisering vil si å bruke z-skår-formelen nevnt i 3.3.2 under innlesning av data. Utifra tabellen virker det som at z-skår-normalisering gir best resultat for alle modellene, mens origo-normalisering gir gode resultater for noen av modellene.

	ANN	RNN	CNN	TCN
Ingen normalisering	10%	4%	4%	17,9
origo-nomralisering	82,9%	70%	4%	88,6
z-skår-normalisering	88,8%	88,8	92,3	91,4%

Tabell 4.4

Figur 4.2 viser visuelt hvordan de forskjellige normaliseringsmetodene påvirker dataene.



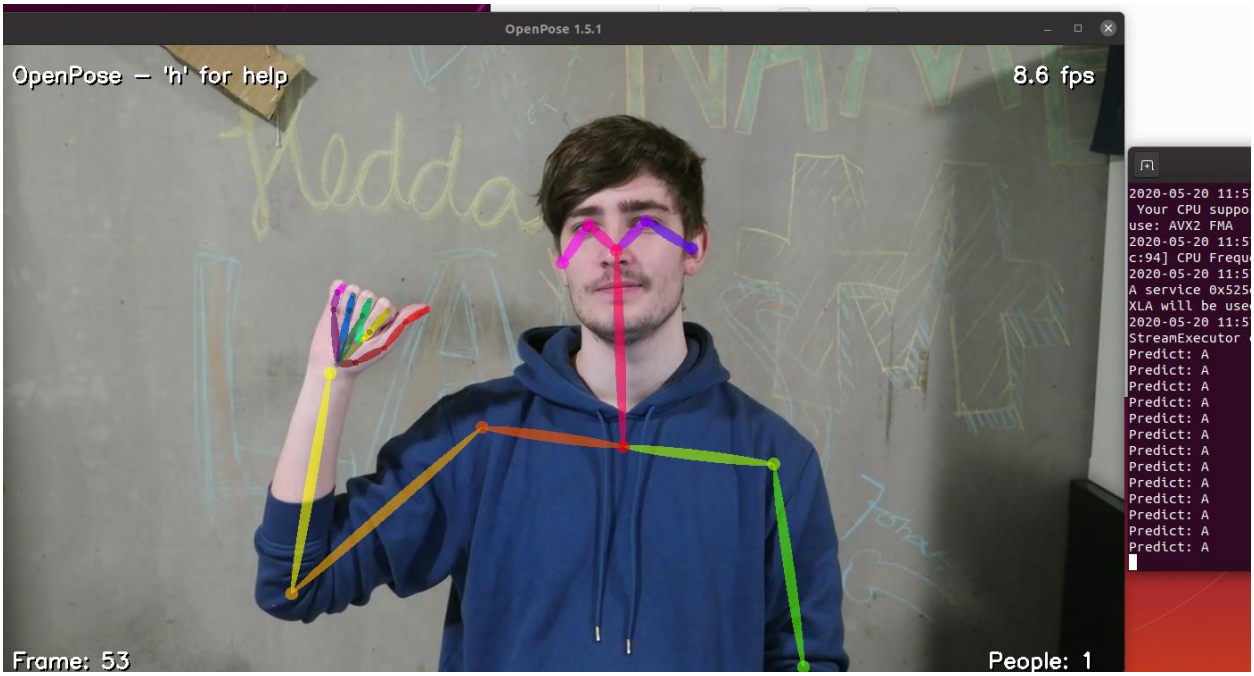
Figur 4.2

Resultatene under er et forsøk på å simulere at Openpose ikke hadde avvik når dataene ble innsamlet. Avvikene til valideringsdataene og treningsdaten ble fjernet og det ble testet om dette påvirket valideringsnøyaktigheten. Mengden data som ble testet på ble da mindre siden avvikene ble fjernet. Testen ble kjørt i 100 epoker på ANN-modellen.

ANN	Traindata	Valdata
Fjernet avik på valdata	97%	90%
Ikke fjernet avik på valdata	97,1%	90%

Tabell 4.5

Det er blitt laget et enkelt program som bruker ANN-modellen beskrevet over i Tabell 4.2, som samarbeider med Openpose for å gjette hvilke tegn som vises i sanntid. Et webkamera sender video til Openpose som lager json-filer og disse blir matet inn i et python-program som skriver ut hvilket tegn personen i videoen holder opp. Dette programmet ble laget for å vise og teste at modellen virker og at det er enkelt å implementere modellene.
Bildet viser et eksempel på kjøring av programmet.



Figur 4.3

Et av målene med oppgaven var å kunne klare å oversette hele ASL-alfabetet ved bruk av maskinlæring. I tillegg til dette var det også et mål å kunne oversette enkle ord eller uttrykk. Som det fremgår av resultatene har målet om oversetting av alfabetet blitt nådd, og da oversettingen av dynamiske tegn var vanskelig å få til ble hovedfokuset rettet mot dette. Enkle

ord på tegnspråk er dynamiske og det var da viktig å oppnå et bra resultat på de dynamiske bokstavene for å kunne jobbe videre med dette.

Timeregnskap

Ligger vedlagt timeregnskap for gruppemedlemmene.

Kapittel 5: Diskusjon

I dette kapittelet vil resultatene og fremgangsmåten både begrunnes og diskuteres. Systemets styrker og svakheter vil bli belyst, samt hvordan svakhetene kunne blitt minsket. Prosjektets måloppnåelse skal også diskuteres i forhold til målsettingen, problemstilling og resultater. Etter intervjuet med tegnspråkstolken Anne Malene Wassmo, virket oppgaven vanskeligere enn forventet, men var også til stor hjelp når det kom til løsing av oppgaven. Intervjuet hjalp oss med tanke på valg av teknologi og metode.

I tabell 4.1 ser vi at begge ANN-modellene viser høyt resultat på trenings- og testdata. Det som skiller de to modellene er nøyaktigheten på valideringsdataen, og man ser at nøyaktigheten på valideringsdataen er betydelig lavere for Mediapipe enn for Openpose. Mye tid ble brukt til å prøve å forbedre modellen som brukte data fra Mediapipe. Ettersom Mediapipe bruker mindre prosessorkraft for å produsere data fra videoer, ville det vært optimalt å kunne bruke Mediapipe. Tiden som ble brukt på Mediapipe ga ikke ønsket resultat og det ble enighet om å fokusere på Openpose ettersom det ga gode resultater. En grunn til at Mediapipe ikke oppnådde like gode resultater som Openpose kan være på grunn av nøyaktigheten på sporingen. Etter å ha sett på sporingen til både Mediapipe og Openpose i sanntid virket det som at Openpose klarte bedre å spore leddene i hånda, mens Mediapipe slet mer med å finne de. Dataene produsert av Openpose vil da være mye mer nøyaktig og inneholde mindre usikkerhet enn dataen fra Mediapipe. Dette kan være en av grunnene til dårlige resultater fra Mediapipe.

Som vist i tabell 4.2 ser man at oversetting av ASL-alfabetet ble oppnådd med høy nøyaktighet. Man ser også fra figur 4.1 at CNN-modellen klarer både statiske og dynamiske bevegelser hvor de dynamiske bevegelsene er spesielt viktig for å kunne oversette tegnspråk. Da CNN- og RNN-modellen trener på ti og ti bilder per iterasjon, krever de også ti bilder for hvert tegn de skal oversette. Dette betyr at valideringsdataen også må deles opp i biter på ti for å kunne oversettes av modellen, noe som fører til færre valideringer totalt sett. Av den grunn har disse modellene ti ganger færre valideringer enn de to andre modellene og derfor er ikke valideringsnøyaktigheten like pålitelig. Valideringsdataen inneholder cirka 3000 bilder, noe som betyr at ANN- og TCN-modellen valideres på 3000 ganger mens CNN- og RNN-modellen valideres bare 300. Fordelen med å trene modellene på ti og ti bilder er at de lettere klarer å se bevegelsen til tegnet fordi det er mer data per gjetning. Dette gjenspeiles i heatmapet i figur 4.1 der man ser at nøyaktigheten til de dynamiske tegnene **J** og **Z** er meget høy. De andre modellene oppnår bare mellom 50 og 70 % nøyaktighet på disse tegnene, noe som er vesentlig dårligere enn CNN-modellen. Dette er grunnen til at CNN-modellen oppnår høyere valideringsnøyaktighet enn de andre modellene. Selv om CNN-modellen presterte bedre på **J** og **Z** med bruk av denne metoden, gjorde ikke RNN-modellen det. I tillegg til at den slet med å identifisere **T**, var det også den modellen som fikk dårligst valideringsnøyaktighet. RNN-modellens virkemåte, beskrevet i kapittel 2.2.2, har den ulempen at den bruker en del minne under trening og trener derfor saktere. Dette medførte at RNN-modellen ble kjørt færre ganger og ble testet med færre parametervariasjoner enn de andre, noe som kan ha ført til at modellen oppnådde lavere nøyaktighet. Spesielt med RNN er at nettverket husker egenskaper med

dataen fra tidligere iterasjoner og epoker, av den grunn trenger ikke modellen nødvendigvis å bruke ti bilder. Det som derfor kunne blitt testet mer, er å trene nettverket med ett og ett bilde for å sjekke om lignende eller bedre nøyaktigheter kunne blitt oppnådd.

En metodene for å prøve å forbedre resultatene til de forskjellige modellene, var å fjerne avvik slik som vises i figur 3.4. Figuren viser at det er en del støy i bildet som blir fjernet. Teorien var at denne støyen bestod av unøyaktige målinger og påvirket modellene negativt. I tabell 4.3 ser vi at fjerning av avvik kun påvirker ANN-modellen positivt. RNN, CNN og TCN blir påvirket negativt av dette. Dette viser at teorien ikke var helt korrekt. Redusert mengde treningsdata kan være en av grunnene til at modellene blir negativt påvirket. En annen grunn kan være at sekvensen til dataen blir forandret. Det kan også hende at denne støyen er noe modellen må lære seg fordi valideringsdataen også inneholder denne støyen, men som man ser i tabell 4.5, har ikke fjerning av avvik på valideringsdataen noen synlig innvirkning på ANN-modellen. Dette kunne også blitt testet på de andre modellene.

I tabell 4.4 ser man at normalisering kan være den viktigste metoden for å oppnå gode resultater. Vi ser også ut i fra tabellen at de forskjellige metodene presterer ulikt. Metodene som blir brukt påvirker dataene på forskjellige måter; synliggjort i figur 4.2. Uten å normalisere ser det ut til å ikke være noe mønster i dataene, mens med de to andre normaliseringsmetodene ser man et visst mønster. Grunnen til at z-skår-normalisering gir så gode resultater kan være fordi normaliseringen gir tall som er lettere å trene på for valgte maskinlæringsmodeller. Uten normalisering av data brukes tall mellom null og 2000, men med z-skår-normalisering brukes tall mellom null og fire, noe som ser ut som å være fordelaktig for modellene. Origo-normalisering gir også tall som er lavere, men avstanden mellom punktene forblir den samme. Det som er fordelen med denne normaliseringen er at punktene samles rundt origo, noe som gjør dataene mindre spredt og mer lesbar. Dette er også en av fordelene med z-skår-normalisering, men z-skår-normalisering lagrer håndens relative størrelse samtidig som den bruker lave tall. Ulempen med normalisering generelt er at håndens posisjon i videosnuttene forsvinner, noe som kan være relevant for tolkningen av tegnspråk.

Noe som kunne forbedret modellene er et større og bedre datasett. Datasettet som ble laget i dette prosjektet er laget av to personer som ikke har mye erfaring med tegnspråk. Dette førte til at noen tegn ble vist feil, noe som igjen medførte at modellene ikke klarte å oppnå ønsket nøyaktighet på valideringsdatasettet. Noen tegn ble også vist på en slik måte at Openpose og Mediapipe ikke klarte å fange opp hånda i det hele tatt. De to ovenfornevnte feilstegene førte til en forlenget periode med trening av modellene, og kunne blitt unngått med å lage datasettet i samarbeid med for eksempel tegnspråkstolker. Under trening av modellene ble det også funnet ut av at bokstaver i ASL-alfabetet kunne vises på ulike måter. Dette førte også til noen problemer under treningsperioden og kunne også blitt unngått hvis gruppens kunnskap om tegnspråk var større. Måten dette ble fikset på var å tilføye datasettet mer data laget fra nye videosnutter, og feilprodusert data ble slettet. Etter flere iterasjoner med datatilsetning oppnådde modellene høyere nøyaktighet.

Som vist i resultatene er databehandling det som påvirker modellene mest. Både normalisering, fjerning av avvik og strukturering av dataen ga store utslag på resultatene. Med gode nok resultater vil det tenkte sluttproduktet nevnt i oppgaveteksten, kunne bli brukt til samfunnsnyttige tjenester som opplæring og oversetting. Produktet kunne også vært aktuelt som et verktøy for universell utforming. Fra et økonomisk perspektiv kan sluttproduktet spare penger for aktører som har bruk for en tegnspråkstolk. Dette kan bli et av produktets salgspunkter. For å realisere dette må oppgavens løsning jobbes videre på, noe som diskuteres i kapittelet under.

Gruppearbeidet

Gruppen bestod av to medlemmer, og det ble utarbeidet møterollene ordstyrer og referent. Det ble ikke valgt noen leder av gruppen ettersom den bestod av to medlemmer. Gruppen valgte å fokusere på én oppgave om gangen og videre ble det enighet om hva som var planen videre etter én oppgave var utført. Denne arbeidsmetoden ble valgt fordi det var ukjent hva som var nødvendig for å oppnå gode resultater.
Oppgaven fikk ikke hovedfokus i starten av prosjektperioden på grunn av andre fag. Senere ble det satt større fokus på oppgaven, noe som vises i timeregnskapet. Gruppen jobbet ekstra hardt for å ta igjen tapt tid.

Kapittel 6: Konklusjon og videre arbeid

Det vil i dette kapittelet redegjøres for prosjektets konklusjon basert på resultatene beskrevet ovenfor, samt videre arbeid. Konklusjonene skal trekkes med hensyn på problemstillingen beskrevet i kapittel 1 og kravene stilt i visjonsdokumentet.

Konklusjon

ASL-alfabetet er nok det enkleste å oversette, men inneholder også noen utfordringer som gjenspeiler hvor bra modellene kan klare å oversette andre tegn. Disse utfordringene er tegn med bevegelse og tegn som er vanskelig å skille mellom. Som bevist i resultatene i kapittel 4 og diskutert i kapittel 5, er de fleste modellene i stand til å identifisere hvert enkelt tegn. Dette ble oppnådd med bruk av todimensjonale koordinater, noe som viser at det er mulig å bruke sporing i xy-rommet til å oversette ASL-alfabetet. CNN-modellen klarer i tillegg å se de dynamiske tegnene med høy nøyaktighet og viser at det er mulig å overkomme disse utfordringene. Dette antyder at det vil være mulig å oversette ord fra tegnspråk til tekst.

Når det kommer til valg av sporingsteknologier ble både Openpose og Mediapipe testet mye. Basert på begges installasjonsprosess, datauthenting og resultat har Openpose alltid prestert bedre. Selvom Mediapipe tok mindre diskplass, krevde det flere forhåndsinstallasjoner, kildekoden måtte redigeres for å hente ut keypoints og sporingen virket generelt dårligere enn Openpose. Openpose ser derfor ut som det beste teknologivalget til lignende oppgaver, noe som også gjenspeiles i resultatene i tabell 4.1. Andre teknologier som oppnår lignende nøyaktighet som Openpose vil være relevante alternativer.

Som nevnt i diskusjonskapittelet er databehandling kanskje det viktigste for å oppnå gode resultater, noe som også indikeres i andre rapporter.[36] Som resultatene antyder fungerer z-skår-normalisering best for datasettet brukt i dette prosjektet, da alle modellene presterer best med denne normaliseringen. Dette tyder også på at modellene lettere oppnår bedre nøyaktighet med de lave normaliserte tallene. Fjerning av avvik var noe som virket både positiv og negativt for modellene. Bruken av dette bør derfor testes for valgt modell da det kan øke nøyaktigheten.

Videre arbeid

Det viktigste arbeidet videre vil være å oversette lette ord og uttrykk. For å oppnå dette bør hele overkroppen spores, noe som er mulig i Openpose ettersom den tilbyr full kroppssporing. Data for det nye datasettet må sannsynligvis bli laget da det finnes svært lite data av denne typen. Hvis et datasett må produseres, kan det være lurt å bruke personer som kan tegnspråk for å få god kvalitet på datasettet. Strukturering av datainnsamlingen vil være viktig for å lage et system som kan ta inn nye ord.

Teknologi er noe som konstant utvikles og forbedrer seg. Det vil derfor være lurt å se etter nye måter å spore kroppen på. Andre måter å løse denne oppgaven på kan være å bruke “action predictions”, der video er det eneste som trengs. Nye maskinlærings algoritmer og modeller kan oppnå gode resultater og vil være relevant å teste.

Anbefalinger for utførelse av lignende prosjekter kan være:

- Datasett kan ta lang tid å lage og behandle, derfor er det nødvendig å lage scripts og lignende for å automatisere prosessen.
- Valideringsdata burde være fra en helt annen kilde enn treningsdata.
- Se på metoder fra andre maskinlæringsprosjekter for å få ideer om hvordan man kan løse eget prosjekt.
- Om treningsdataen er selvprodusert, sjekk nøye om den er tilfredsstillende og om den fungerer som den skal.
- Bruk charts for å lettere analysere dataen og nøyaktighet.
- Start med en plan for mappestruktur

Referanser

1. Språkrådet, “Hvor mange snakker norske tegnspråk?” Lesedato 06.05.20 https://www.sprakradet.no/Spraka-vare/Tegnsprakteiknsprak/Ofte-stilte-sporsmal-om-tegnsprak/hvor-mange-snakker-norsk-tegnsprak/

2. Døveforbundet, “Likepersonarbeidet i Norges Døveforbund” Lesedato 14.05.20 https://www.doveforbundet.no/likeperson/arbeidet

3. “Sign language” Lesedato 26.04.20 https://en.wikipedia.org/wiki/Sign_language

4. Eivind Alfsvåg Johansen 18.02.20 Intervju med Anne Malene Wassmo

5. “American Sign Language” Lesedato 26.04.20 https://www.nidcd.nih.gov/health/american-sign-language

6. “Asl_aplhabet_gullaudet” hentedato 26.04.20 https://upload.wikimedia.org/wikipedia/commons/c/c8/Asl_alphabet_gallaudet.svg

7. mars 2017. “What is Machine Learning? A definition” Lesedato 27.04.20 https://expertsystem.com/machine-learning-definition/

8. Chen, James 19 okt. 2019. “Neural Network” Lesedato 28.04.20 https://www.investopedia.com/terms/n/neuralnetwork.asp

9. “Neural Network” hentedato 28.04.20 https://en.wikipedia.org/wiki/Neural_network

10. “Keras - Dense Layer” Lesedato 29.04.20 https://www.tutorialspoint.com/keras/keras_dense_layer.htm

11. West, Mark 20.august 2019. “Explaining Recurrent Neural Networks” Lesedato 02.05.20 https://www.bouvet.no/bouvet-deler/explaining-recurrent-neural-networks

12. Hoffmann, Garret 11. Januar 2018. “Introduction to LSTMs with Tensorflow” Lesedato 05.05.20 https://www.oreilly.com/content/introduction-to-lstms-with-tensorflow/

13. “Whats the difference between LSTm and GRU” lesedato 06.05.20 https://www.quora.com/Whats-the-difference-between-LSTM-and-GRU

14. “Convolutional Neural Network” lesedato 03.05.20 https://en.wikipedia.org/wiki/Convolutional_neural_network

15. Colin Lea Michael D. Flynn Rene Vidal Austin Reiter Gregory D. Hager 16. November 2016 “ Temporal Neural Networks for action segmentation and segmentation” lesedato 09.05.20 https://arxiv.org/abs/1611.05267

16. Shaojie Bai, J. Zico Kolter, Vladlen Koltun 4. Mars 2018 “An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling” lesedato 09.05.20 https://arxiv.org/abs/1803.01271

17. “Activation Function” lesedato 27.05.20 https://en.wikipedia.org/wiki/Activation_function

18. Brownlee, Jason, July 5, 2019, “How to Configure Image Data Augmentation in Keras”, lesedato 12.02.2020 https://machinelearningmastery.com/how-to-configure-image-data-augmentation-when-training-deep-learning-neural-networks/

19. Brownlee, Jason, July 5, 2019, “Overfitting and Underfitting with machine learning algorithms” https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/

20. Brownlee, Jason, August 12, 2019, “Overfitting and Underfitting With Machine Learning Algorithms” lesedato 10.02.20 https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/

21. “Kinect” lesedato 03.02.20 https://en.wikipedia.org/wiki/Kinect

22. “Stero Vision” lesedato 20.04.20 https://www.cse.unr.edu/~bebis/CS791E/Notes/StereoCamera.pdf

23. “Dlib” lesedato 27.01.20 http://dlib.net/

24. “Opoencv” lesedato 26.01.20 https://opencv.org/

25. “Openpose” lesedato 25.01.20 https://github.com/CMU-Perceptual-Computing-Lab/openpose

26. “Mediapipe” lesedato 25.01.20 https://github.com/google/mediapipe/

27. “Manomotion” lesedato 14.04.20 https://www.manomotion.com/

28. “Usense” lesedato 14.04.20 https://en.usens.com/

29. Elena Ceseracciu, Zimi Sawacha, Claudio Cobelli, Christof Markus Aegerter, 2014 Mar 4, “Comparison of Markerless and Marker-Based Motion Capture Technologies through Simultaneous Data Collection during Gait: Proof of Concept” lesedato 13.05.20 https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3942307/

30. Katsu Yamane, Jessica K. Hodgins, 2009, “Simultaneous tracking and balancing of humanoid robots for imitating human motion capture data” lesedato 10.05.20 https://www.semanticscholar.org/paper/Simultaneous-tracking-and-balancing-of-humanoid-for-Yamane-Hodgins/1ac8834aaa852507462a692ca5e7f036c47375e4

31. “Tensorflow” helesedatontedato 10.05.20 https://www.tensorflow.org/

32. “Keras” lesedato 10.05.20 https://keras.io/

33. “Pandas” lesedato 10.05.20 https://pandas.pydata.org/docs/#

34. Yajie Miao, Mohammad Gowayyed, Florian Metze, 18 Oct 2015, “EESEN: End-to-End Speech Recognition using Deep RNN Models and WFST-based Decoding” lesedato 10.03.20 https://arxiv.org/pdf/1507.08240.pdf

35. Shaojie Bai, J. Zico Kolter, Vladlen Koltun, Thu, 19 Apr 2018, “An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling”, lesedato 14.03.20 https://arxiv.org/abs/1803.01271

36. Sang-Ki Ko, Chang Jo Kim, Hyedong Jung, Choongsang Cho, 21 Jun 2019, “Neural Sign Language Translation based on Human Keypoint Estimation”, lesedato: 14.05.20 https://arxiv.org/abs/1811.11436

Vedlegg

Det ble valgt å ikke inkludere kravdokumentasjon, da løsningen utviklet i prosjektet ikke er et større system bestående av backend og frontend med flere moduler. Derfor vil heller ikke systemdokumentasjonen inneholde informasjon om lignende tema.

Innhold

Møteinnkallinger med referat

Møteinnkalling til oppstartsmøte

Følgende personer innkalles
- Holt, Tomas
- Jacobsen, Sigurd Løite
- Johansen, Eivind Alfsvåg

Dato: 16.01.20
Tid: 14:30
Sted: IT-bygget

Det kalles med dette inn til oppstartsmøte for bacheloroppgaven om tegnspråk

Saksliste

1. Åpning av møte
2. Godkjenning av saksliste
3. Møteplan
4. Inndeling av prosjektet i deler
5. Framdriftsplan og utviklingsprosess
6. Krav til dokumentasjon
7. Retningslinjer for vurdering
8. Delområde for bacheloroppgave
9. Tegnspråk
10. Valg av språk
11. Eventuelt

Med hilsen Sigurd Løite Jacobsen og Eivind Alfsvåg Johansen

Møtereferat Oppstartsmøte

Dato: 16.01.20
Tid: 14:30
Sted: IT-bygget
Til stede: Tomas Holt (veileder), Sigurd Løite Jacobsen, Eivind Alfsvåg Johansen
Frafall: Ingen
Ordstyrer: Eivind Alfsvåg Johansen

Møteplan

Ble enighet om å ha møter etter studentenes eller veileders behov.

Inndeling av prosjektet i deler

Skrive logg underveis
Oppdelingen av prosjektet ble som følgende:

- Oppstartsfase : research, se på tidligere arbeid
- Midtfase: Implementere noe, maskinlæring, håndtracking
- Sluttfase: sluttrapport

Framdriftsplan og utviklingsprosess

Det ble enighet om å diskutere utviklingsprosessen i sluttrapporten. Reseach-preget prosjekt. Legge vekt på researchen. Hovedrapporten er en stor del av prosjektet.

Krav til dokumentasjon

Det ble enighet om at kravene til dokumentasjon er hovedrapport, visjonsdokument, dokumentasjon på det vi lager(installasjon, brukerveiledning)

Retningslinjer for vurdering

Førstinntrykket på hovedrapporten er veldig viktig. Få frem negative og positive sider på det vi gjorde/fant ut av.

Delområde for bacheloroppgave

Maskinlæring. Værtfall I starten

Tegnspråk

Prefererer bildeanalyse ovenfor marker tracking. Tidligere har det blitt oppnådd gode resultater med Leadmotion. Hele alfabetet har blitt klart å oversette. Kinect kan brukes for å får med hele bevegelser. Teknologier som kan brukes: Mimic og Fclone - ansiktsgjenkjenning som gir markørdata i ansiktet. Ble enige om å gjøre mye research på hvilke teknologier som brukes og samtidig lære oss litt om hvordan tegnspråk fungerer og hva som er relevant å tracke.

Valg av språk

Rapport på norsk og kode på engelsk.
Brukerveiledning / installasjon på engelsk.

Eventuelt

Se på Amerikansk database med tegnspråk?

Møteinnkalling til veiledningsmøte 1

Følgende personer innkalles:

- Holt, Tomas
- Jacobsen, Sigurd Løite
- Johansen, Eivind Alfsvåg

Dato: 04.02.20
Tid: 1400
Sted: IT-bygget

Det kalles med dette inn til veiledningsmøte for bacheloroppgaven om tegnspråk

Saksliste

1. Åpning av møte
2. Godkjenning av saksliste
3. Valg av problemstilling
4. Gjennomgang av visjonsdokument
5. Videre arbeid og det vi har funnet ut av
6. Eventuell sak
7. Godkjenning av møtereferat

Med hilsen Sigurd Løite Jacobsen og Eivind Alfsvåg Johansen

Møtereferat veiledningsmøte 1

Dato: 04.02.20
Tid: 1400
Sted: IT-bygget
Til stede: Tomas Holt (veileder), Sigurd Løite Jacobsen, Eivind Alfsvåg Johansen
Frafall: Ingen
Ordstyrer: Eivind Alfsvåg Johansen

Valg av problemstilling

Må spisses inn sånn at den kan svares på mot slutten av oppgaven

Visjonsdokument

Lage et system som kan ta opp tegnspråk, visualisere det, oversette det. Det primære er å få teknologien ut. Under beskrives hva som må skrives om under de forskjellige delkapitlene i visjonsdokumentet

3.3 – tenk på brukeren, brukere: omsorgspersoner, opplæringssenter, kan også være enkeltpersoner hvis produktet blir bra nok. Må da fungere på mobiltelefonen. Se på hvilke miljøer produktet skal passe inn i.

4.1 – hjelpemiddel, for døve og de som er rundt.

5. – Funksjoner produktet skal ha. Ikke gå så mye i detalj.

6- Hva slags hardware, utelse? Responsiv, kamera, f.eks.

Prosjektet er veldig forskningsorientert.

Det vi har fått til og videre arbeid

Det viktigste er å få til en oversettelse.

Sammenligne ansikttracking-bibliotekene og andre bibliotek mot hverandre i rapporten.

Se på intels kameraer.

Forskjellige kamera på mobil – kan nevnes som videre arbeid i rapporten.

Videre arbeid

Teste libraries og produktet, med fokus på håndbevegelser.

Bruke googles hånd-tracker for å få til det spanjolene gjorde?

Bruk prosjektet som gjorde dette før som referanse.

Møteinnkalling til veiledningsmøte 2

Følgende personer innkalles:

- Holt, Tomas
- Jacobsen, Sigurd Løite
- Johansen, Eivind Alfsvåg

Dato: 27.03.20
Tid: 13:45
Sted: meet.jit.si

Det kalles med dette inn til veiledningsmøte for bacheloroppgaven om tegnspråk

Saksliste

1. Åpning av møte
2. Godkjenning av saksliste
3. Fremdrift
4. Videre arbeid
5. dokumentasjon(systemdokumentasjon, kravdokumentasjon)
6. eventuelt

Med hilsen
Sigurd Løite Jacobsen og Eivind Alfsvåg Johansen

Referat Veiledningsmøte 2

Dato: 27.03.20
Tid: 13:45
Sted: meet.jit.si
Til stede: Tomas Holt (veileder), Sigurd Løite Jacobsen, Eivind Alfsvåg Johansen
Frafall: Ingen
Ordstyrer: Eivind Alfsvåg Johansen

Fremdrift

Vi har fått til en modell som klarer ca. 80% acc på både openpose og mediapipe, der openpose har fått til litt bedre predictions, med bruk av data augmentation og et CNN(convolutional neural network).

Videre arbeid

Som videre arbeid ble det diskutert om det kunne vært mulig å få dette til i realtime og/eller raskere. Men det ble konstatert at det viktigste er å få til å predikere riktig tegn.

dokumentasjon(systemdokumentasjon, kravdokumentasjon)

Det ble diskutert hva systemdokumentasjonen skulle inneholde. Her ble vi enige om at mange av punktene skulle droppes og at det viktigste er at man får frem hvordan koden er bygd opp og hvordan man bruker den. Når det gjaldt kravdokumentasjon ble vi enige om at vi skulle argumentere for at det ikke er hensiktsmessig å skrive kravdokumentasjon, da dette prosjektet er et forskningsprosjekt og ikke et systemutviklingsprosjekt.

Eventuelt

Sende kontrakt om hvem som har hvilke rettigheter

Møteinnkalling til veiledningsmøte 3

Følgende personer innkalles:

- Holt, Tomas
- Jacobsen, Sigurd Løite
- Johansen, Eivind Alfsvåg

Dato: 17.04.20
Tid: 13:00
Sted: meet.jit.si

Det kalles med dette inn til veiledningsmøte for bacheloroppgaven om tegnspråk

Saksliste

1. Åpning av møte
2. Godkjenning av saksliste
3. Fremdrift
4. Usens og Manomotion
5. Videre arbeid
6. eventuelt

Med hilsen
Sigurd Løite Jacobsen og Eivind Alfsvåg Johansen

Møtereferat veiledningsmøte 3

Dato: 17.04.20
Tid: 13:00
Sted: meet.jit.si
Til stede: Tomas Holt (veileder), Sigurd Løite Jacobsen, Eivind Alfsvåg Johansen
Frafall: Ingen
Ordstyrer: Eivind Alfsvåg Johansen

Fremdrift

Ta med vurdering av manomotion i rapport. Bør være et åpent produkt pga da er det mest bærekraftig.

Videre arbeid

Begynne å fylle inn små ting i rapporten.
Få bedre ytelse med openpose?
Se videre på RNN-nettverket.
Se på ytelse opp mot nøyaktighet.
https://github.com/CMU-Perceptual-Computing-Lab/openpose#speeding-up-openpose-and-benchmark
https://github.com/CMU-Perceptual-Computing-Lab/openpose/blob/master/doc/speed_up_openpose.md
https://docs.google.com/spreadsheets/d/1-DynFGvoScvfWDA1P4jDInCkbD4lg0IKOYbXgEq0sK0/edit#gid=0

Få med forbedringspotensialer i rapport.

Jobb mer videre med modellen vi har, ikke se på nye SDK, blir litt for mye arbeid.
Fortell om hvorfor openpose er bra i rapporten selvom den er treg.
Nevn i rapporten løsning med to kamera, 3D keypoints i openpose.

Møteinnkalling til veiledningsmøte 4

Følgende personer innkalles:

- Holt, Tomas
- Jacobsen, Sigurd Løite
- Johansen, Eivind Alfsvåg

Dato: 06.05.20
Tid: 1400
Sted: meet.jit.si

Det kalles med dette inn til veiledningsmøte for bacheloroppgaven om tegnspråk

Saksliste

1. Åpning av møte
2. Godkjenning av saksliste
3. Fremdrift
4. Videre arbeid
5. Spørsmål til rapporten

a. Rollefordeling
b. Spør veilerder om å bruke data fra youtube
c. Spør veileder om admin og ingeniør resultater
d. Strukturering av kapittel 3
e. Om vi kan send hovedrapporten til veileder
f. Hvordan det er å skrive “vi”.
g. Kilder wikipedia, hvordan kildene skal vises(footnote)

6. eventuelt

Med hilsen
Sigurd Løite Jacobsen og Eivind Alfsvåg Johansen

Referat for veiledningsmøte 4

Dato: 06.05.20
Tid: 1400
Sted: meet.jit.si
Til stede: Tomas Holt (veileder), Sigurd Løite Jacobsen, Eivind Alfsvåg Johansen
Frafall: Ingen
Ordstyrer: Eivind Alfsvåg Johansen

Fremdrift

Vi har fått til ganske god nøyaktighet på alle modellene

Videre arbeid

Jobb videre med og ferdigstill rapporten og andre dokumenter

Spørsmål til rapporten

1. Rollefordeling

a. Nevn litt kjapt, vi har begge utført alle rollene

2. Spør veilerder om å bruke data fra youtube

. Ta med referanse til video som ble brukt til å generere testdata.

3. Spør veileder om admin og ingeniør resultater

. Ingeniørfaglig resultater: det vi har oppnådd som ingeniører, det vi har laget. Det faglige,

a. Administrative resultater: hvordan prosjektet ble administrert, har ting funka, timeregnskap, hvordan vi har gått fram når vi jobba. Vurder det vi gjort selv, noe som kunne gjort annerledes.

4. Strukturering av kapittel 3

. Ikke så nøye.

5. Om vi kan sende hovedrapporten til veileder

. Nei. still heller konkrete spørsmål

6. Hvordan det er å skrive “vi”.

. Helst ikke, med mindre det er snakk om personlige opplevelser

7. Kilder wikipedia, hvordan kildene skal vises(footnote)

. Wikipedia er greit, bruk footnote

Timelister og logg

Timeliste og logg Eivind Alfsvåg Johansen

Eivind		
Date:	Timer:	Breskrivelse
13-Jan-2020	2	Møteinnkalling
19-Jan-2020	3	Research
21-Jan-2020	5.5	Visjons dokument og research
27-Jan-2020	3	Reserch og kontaktet tegnspråks lærer
29-Jan-2020	2	Research og interview med Anne Malene
31-Jan-2020	8	Research
14-Feb-2020	6	Research
17-Feb-2020	8	Ubuntu problemer og mediapipe install
18-Feb-2020	8	Testing av gpu og cpu med hand tracking
19-Feb-2020	8	Testing av openpose og installasjon
20-Feb-2020	8	Laging av datasett for å teste og lage ml algorytme
25-Feb-2020	8	Jobbet med å få ut data fra mediapipe
26-Feb-2020	8	Jobbet med å få ut data fra mediapipe fikk det ut
27-Feb-2020	8	Jobbet med mediapipe
28-Feb-2020	8	Fikk mediapipetil å printe json filer og openpose
2-Mar-2020	10	Lastet inn data til makinlæring
3-Mar-2020	10	lager maskin algorytme
4-Mar-2020	8	maskin læring
5-Mar-2020	8	maskin læring
6-Mar-2020	10	jobbet med lagene i maskinlæring modellen
9-Mar-2020	10	maskin læring
10-Mar-2020	10	makinlæring algo 97% staving
23-Mar-2020	10	dataaugmentation
24-Mar-2020	10	Lage større datasett og få venstre hånd med
25-Mar-2020	10	Jobbet med algoryten til makinllærings prosjektet
26-Mar-2020	10	Jobbet med å forbedre algorytmen
27-Mar-2020	10	Møte og justere modellen og skaffe data for å dobbeltsjekke
30-Mar-2020	10	fikk 95 % val acc og testing på videos på youtube ble 73%
31-Mar-2020	10	Hadde feil i data ryddet opp
1-Apr-2020	10	Jobbet med å få tensorflow til å virke på gpu og forbedret modellen
2-Apr-2020	10	Jobbet med å få modellen til å finne T
3-Apr-2020	10	Prøvde fosjellige dense lag og lagde slik at all data er lik mengde

14-Apr-2020	10	Lagret modellen og kjørte det realtime
15-Apr-2020	10	Laget en LSTM modell
16-Apr-2020	10	Lagde en RNN modell
17-Apr-2020	8	Jobbet med modellen og hadde møte med prosjekt veileder
20-Apr-2020	8	jobbet med å finne måter å gjøre openpose raskere
21-Apr-2020	10	Jobbet med hovedrapporten og RNN modellen
22-Apr-2020	10	Testet ut forsjellige modeller
23-Apr-2020	8	Testet forsjellige dense lag og forsjellige mengder frames i hver prediction
24-Apr-2020	10	Lagde ny modell TCN
27-Apr-2020	10	Skrev på rapporten og lagret modellen til TCN
28-Apr-2020	8	Skrev på rapporten
29-Apr-2020	8	jobbet med rapport teori
30-Apr-2020	8	jobbet med rapport teori
1-May-2020	8	jobbet med rapport teori
4-May-2020	8	jobbet med rapporten metode og lagde et par videoer med J og Z
5-May-2020	8	jobbet på rapporten og lagde CNN i egen jupyter og ryddet litt opp i koden
6-May-2020	8	Lagret modellene på en annen måte og skrev på rapporten om valg av teknologi
7-May-2020	8	skrev på rapport om metode
8-May-2020	10	Skrev på rapport
10-May-2020	8	Skrev på rapport resultat
11-May-2020	8	Skrev på rapport
12-May-2020	8	Skrev på rapport
13-May-2020	8	Skrev på rapport
14-May-2020	11	Skrev nesten ferdig rapport
15-May-2020	10	Installasjone guide og andre dokumenter
16-May-2020	6	Ferdigstilling av rapport
17-May-2020	6	Ferdigstilling av rapport
18-May-2020	6	Ferdigstilling av rapport
19-May-2020	6	Ferdigstilling av rapport
sum:	504.5	

Logg:
27.01.20

Jobbet med å lage spørsmål og komme i kontakt med tegnspråks lærer for spørsmål.

Har sett på forskjellige måter vi kan oppnå kropps tracing med bare ett kamera. Det virker vanskelig å få ut 3d data ut av video, men ser ut som noen har klart det uten at det var mye informasjon om det. Bare

att de viste 3d tracking over noen sports videoer ser ut som de bruker flere videokameraer (CVSSP Reaserch).

29.01.2020

Jobbet en time med å finne måter å tracke kropp og hender. Fant flere måter å gjøre det på den vanligste er dybde kamera eller RGB-D cameras som det også kalles. Denne blir mye brukt. Den har noen problemer den er litt dyr, ikke alle har tilgang til dybdekamera, kan få forstyrrelser fra sol og andre dybdekmeraer. Stereo kamera kan bli brukt til å tracke hender likt som dybdekamera virker også har ikke problemet med forstyrrelser fra sol og andre kameraer, ikke mange som har tilgang til stereo kamera. Vanlig kamera som er på mobil kan brukes til å estimere kropp vet enda ikke heilt om det kan brukes til estimere hand tracking.

https://arxiv.org/pdf/1704.02201.pdf dybde sensor

https://arxiv.org/pdf/1610.07214.pdf stereo kamera

https://www.youtube.com/watch?v=m3KG_Z0P_nU kamera vanlig rgb

Trenger fortsatt ansikts tracking finn tekster for det.

Hadde interview med Anne Malene fikk vite mye om tegnspråk og hvor komplisert det er og hva av kroppen som ble brukt samt at det er lite standarisert.

Skrevet hva jeg fant ut på dockumentet lagt i tegnspråk mappen

31.01.2020

Viola-Jones object detection framework er en måte å finne objekter på som hender ansikt og andre ting i bilde. Kan brukes til å kjøre maskinglærings algorytme på deler av bildet.

Dlib er et library som tar ca 30 ms

Clm-Framework verre licence bra tracking bedre enn DLib

https://www.signall.us/product_info/ siggurd fant en løsning som bruker flere kameraer og fargerike hansker til å se hvilke fingrer som er hvilke

17.02.2020

Prøvde å installere media pipe på windows men fant ut at det var dumt ettersom det ikke kunne kjøre på gpu. Hadde alt for mange problemer med å innstallere linux og mestepartten av dagen gikk ut på det.

18.02.2020

Fikk testet mediapipe handtracking var dårligere enn forventet. Den må kjøre på gpu for at det skal være vits å bruke. Hvis hendene beveger seg for fort så vil de bli blurry og den klarer ikke å ta opp dette. Trackingen er lite presis og rister hele tiden.

19.02.2020

Openpose virket bedre enn mediapipe. Estimeringen ristet ikke like mye som mediapipe. Er i tvil om noen av disse kommer til å gi et godt resultat, men vi skal prøve å teste disse mot hverandre og kansje teste de mot å ikke bruke disse(bare bruke video i mskinlæring. Tenker å lage et lite datasettet neste gang med video av alfabetet.

20.02.2020

Dro til Sigurd og lagde noen videoer der han viste forskjellige tegnspråks bokstaver. Dette skal vi bruke i maskinlærings algorytmene vi skal lage.

26.02.2020

Fikk ut data fra mediapipe. Det neste blir å strukturere dataene vi får ut og lable de slik at de kan brukes i maskinlærings algorytmer. Fikk til å kjøre mediapipe med video slik at vi slipper å bruke en virutuel webcam for å kjøre videoene igjennom.

28.02.2020

Fikk mediapie til å printe json filer for hver frame. De blir lagt inn i rett mappe. Lagde shell scripts for å automatisere alt. hvis vi lager større datasett kan vi bare kjøre scriptene så vil det bli gjordt alt av forberedende arbeid.

kan begynne å se på maskin lærings algorytmer neste gang og data augmentation.

02.03.2020

lagde en jupyter notebook og fikk lastet inn dataene etter mye omstrukturering av data.

03.03.2020

Jobbet med maskinlærings algorytrme

04.03.2020

jobbet med maskinlærings modellen

05.03.2020

fortsatte å jobbe med maskinlærings modellen

05.03.2020

testet forsjellige lag

09.03.2020

Jobbet med maskinlæring. Nådde 77 % acc på 100000 epocher. Skal prøve å få tensorflow til å kjøre på gpu slik at det ikke tar så lang tid å kjøre hver gang.

10.03.2020

97% på 1000 epocher. med en bedre algorytme. har kansje dårligt datasett må teste med annen data.

23.03.2020

Fant ut at datasettet vi har er det problemer med overfittiong. Virker veldig dårligt på data som er ikke fra testing eller trening. Vi har laget noen metoder for å endre på dataene slik at de er litt anderleser (Dataagumentation). Disse håper vi skal løse problemet litt, men vi skal også lage litt større datasett forhåpentligvis.

24.03.2020

Jobbet med å lage større datasett og mer varierende. Det viste seg at vi ikke hadde tenkt på om personen brukte høyre eller venstre hand. Vi har nå et datasett til venstre hand. Vi må se på hva vi skal gjøre og om det er muligt å få algoritmen til å virke på begge hvis den vet om det er høyre eller venstre. Tar en del tid å hente data ut fra video men når det er gjordt trenger man ikke å gjøre det igjen.

25.03.2020

jobbet med å fikse noen videoer som vi brukte feil hand på. Prøvde å lage en many to one modell. Fikk ikke til.

26.03.2020

Har en del problemer med at maskinlærings algorytmen blir sittende fast på 0.0541 prosent sjanse. Bare noen verdier får den til å kjøre forbi den og da får vi ca 80% acc. Jobber med å ha en bedre måte å teste dataen på. Noen som ikke er i datasettet som gjør tegnbevegelsene. Testet flere ‘activation’ typer for dense lagene. selu var veldig bra, men må gjøre noen spesielle ting fo

27.03.2020

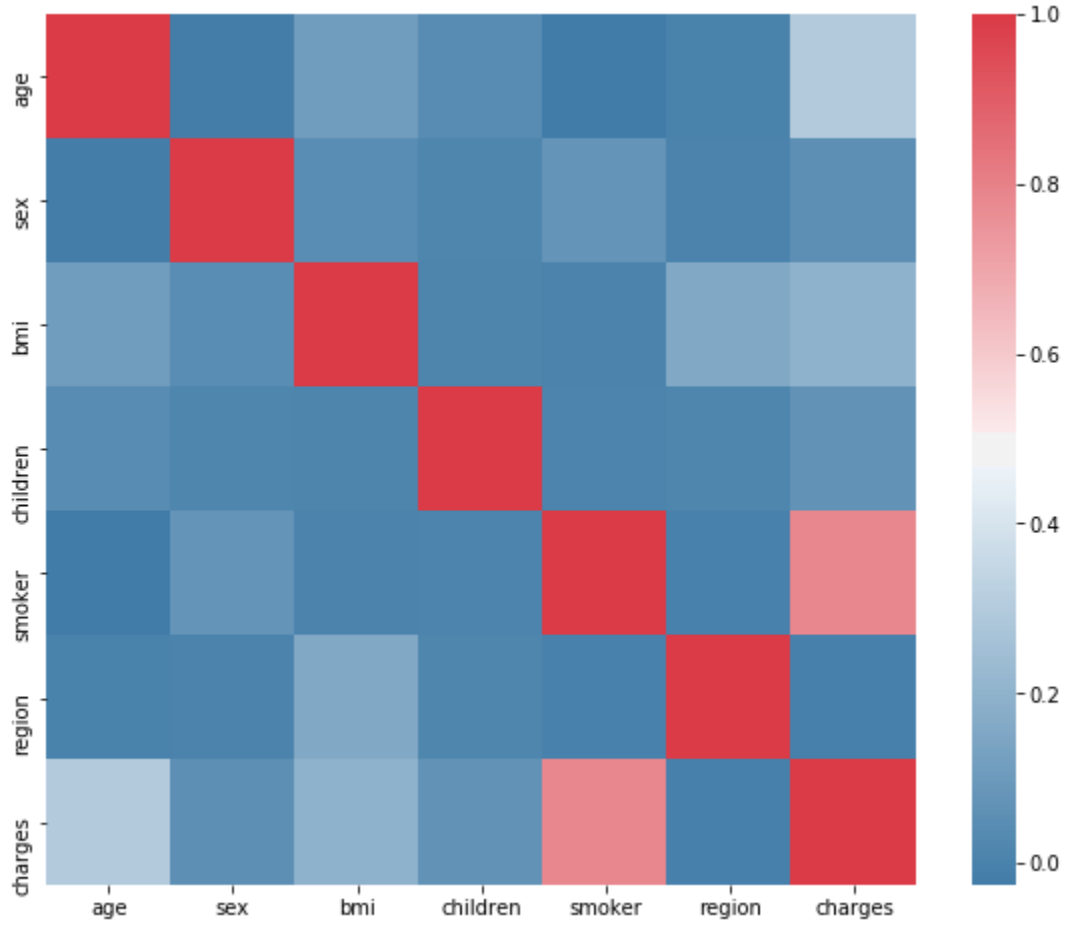
Møte med veileder. Det gikk fint vi skulle fokusere på å forbedre modellen slik at kansje etter påske kunne vi bruke litt kraftigere maskinvare. Justerte modellen og skaffet ekstra data for å dobbeltsjekke om modellen virker.

Til neste uke:

fikse trenings data(remove outliers)

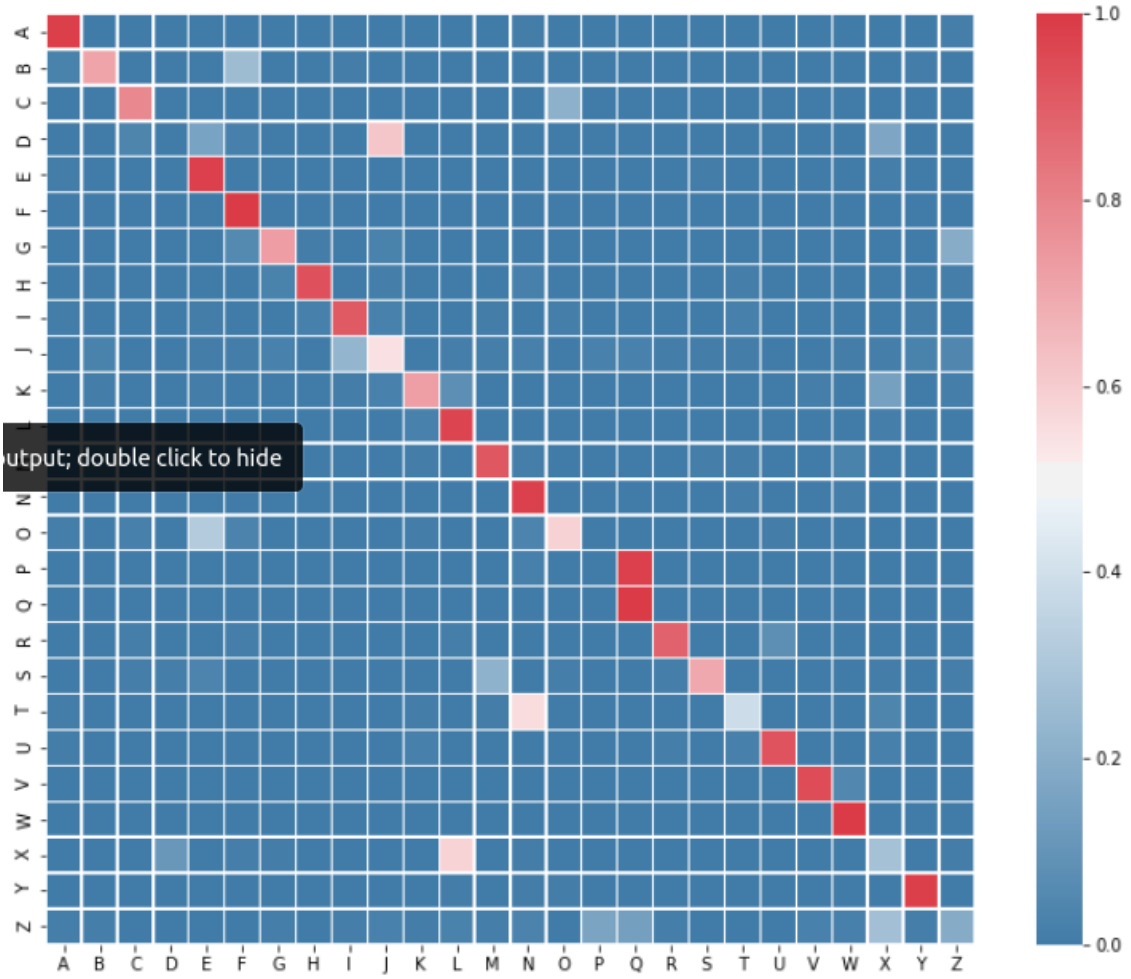
forbedre modell(mer effektiv, og mer acc)

fikse chart



30.03.2020

lagde graf:



Gjorde slik at all data starter i x, y posisjon 0,0 for å normalisere dataen. Vi så at dette økte treffsikerheten til modellen. Vi fjernet også data som var outliers. Det fjernet ca 20 % av dataen vår. Mediapipe viser seg å være veldig dårlig i forhold til openpose. Vi fikk 95% validation acc på modellen på 100 epocher. Den var nærmere 73 % når vi testet den på youtube videoer. Dette skyldes at noen av tegnene gjøres litt forskjelligt. C kan du vise rett fram eller til siden det vil forandre hvordan verdier den får.

31.03.2020

Fant ut at noen av videoene kunne ikke hentes data fra og vi lagde nye slike videoer. Det var også en video som hadde feil tegn. Vi rettet opp i disse feilene. Vi såg også på andre mulige måter å tracke hendene på ios og andorid. Usans og manomotion skal se nermere på disse.

01.04.2020

Jobbet med å finne en modell som er så lik som validation accen. Testet mange forsjellige måter å sette opp modellen.

02.04.2020

Prøvde å finne ut hvorfor T var så vanskelig å gjennkjenne. Fant ut at openpose hadde litt problemer med å se heile bevegelsen. Tegnet er også veldig likt både S og N. Hvis du bare har 2d kordinater så er de egentlig identiske.

03.04.2020

Jobbet med å få treningsmodellen til å gjennkjenne test dataen bedre. Fant ut at å øke dropout hjalp veldig mye.

14.04.2020

Lagret den beste modellen og lagde et program slik at det kan kjøres real time eller med video.

15.04.2020

Jobbet med å lage en ny modell som husker hva forrige state var(LSTM). Får ikke så bra acc får se på hva som kan være grunnen til det. Testet med å fjerne outliers men det ødela modellen kan se på dette senere.

16.04.2020

Gjorde om LSTM modellen til RNN og fikk litt bedre resultat. Ikke så bra at det går ant å bruke. fortsatt ANN modellen som er best.

17.04.2020

Jobbet med modellen. hadde møte med prosjekt leder. fant ut at vi skal jobbe mer med modellen og at vi skulle ta kontakt med uSense og Manomotion og se hva de hadde å by på. vi skulle og finne bencmarks for openpose og se på om det går atn å øke fpsen til openpose og se hvor nøyaktig den blir da.

20.04.2020

Jobbet med å gjøre openpose raskere. Jobbet med at opencv skulle ha cuda og opengl support.

21.04.2020

Jobbet med hovedrapporten skrev hva vi hadde tenkt tiul å skrive slik at når vi begynner å skrive blir det lettere. Jobbet med RNN modellen satt dataen i 30 og 30 arrays. Vi fikk bedre resultat på J og Z men vi sliter fortsatt med accen på modellen frøvde nå å ha GRU LSTM og RNN i samme modell.

22.04.2020

Jobbet med modellen. Dataen er blitt normalisert må en ny måte der vi tar

$$V_x^* = \frac{V_x - \bar{V}_x}{\sigma(V_x)},$$

Vi sender inn ord som data nå. slik at HALLO blir 24 frames med H og 24 frames med A også videre.

Prøvde å forbedre RNN fikk ikke til veldig mye. Prøver nå CNN

23.04.2020

Skrev på rapporten på forord. Testet forskjellig mengde frames 5, 10, 20,30 ,40. Ser at vi får best resultat med 30 frames, men da predicter den bare 1 gang i sekundet. Vi fikk nesten like braa resultat med 10 frames så vi prøver å forbedre denn modelle. Dette vil gjøre slik at vi predicter 3 ganger i sekundet og vi har sett an noen 'snakker' veldig fort med tegnspråk.

24.04.2020

Lagde en modell Med TCN virker bra er ca like bra eller bedre som de andre modellene. Denne modellen ser på de forje lagene og gjør predictions etter det. Den har noen artikler som viser at den slår RNN og LSTN på datasett med veldig optimaliserte modeller.

27.04.2020

Skrevet på hovedraport. om forsjellige kameraer og forjellige metoder å fange bevegelse. Lagret en modell. Vi tror at J og Z har for lite data, skal prøve å lage noen nye videoer.

28.04.2020

skrev på hovedrapporten

29.04.2020

skrev på hovedrapporten

01.05.2020

skrevet på rapporten om teori… som de andre dagene

04.05.2020

skrev på rapport. mye kladd men kan begynne å skrive på metode i mårra. Fikk litt bedre resultat på RNN fortsatt ikke noe å skryte av. lagde en ekstra video av J og Z fikk litt bedre resultater da.

05.05.2020

skrev på rapport om teknologi drøfting

fikk cnn til å gi 95.4 acc

06.05.2020

Lagret modellene med alle lagene slik at du ikke må vite hvor dan modellen er bygget opp for å kjøre den. Skrev på rapporten om valg av teknologi.

07.05.2020

skrev på rapporten om metode. lagde noen charts omn outliers

08.05.2020

Skrev på rapport lagde noen charts

10.5.2020

skrev på rapport gjorde noen tester

fikk 97.8 på cnn modellen

11.05.2020

skrev på diskusjon

14.05.2020

skrev på rapport

Timeliste og logg Sigurd Løite Jacobsen

Sigurd	Timer	Beskrivelse
13.01.20	5	Møteinnkalling og forberedelser
20.01.20	4	Visjonsdokument
21.01.20	8	Research og visjonsdokument
29.01.20	2	Intervju med tegnespråklærer
13.02.20	8	Research om sporingsteknologier
14.02.20	8	Research om sporingsteknologier
17.02.20	8	Installasjon av mediapipe og research
18.02.20	8	installasjon av mediapipe og research
19.02.20	8	Installasjon av openpose, og generering av JSON fra Openpose
20.02.20	9	lagde video av tegnspråk-alfabetet
26.02.20	9	lagde json-fil av punktene på hånda fra mediapipe.
27.02.20	7	lagde json-fil av punktene på hånda fra openpose
28.02.20	9	Lagde og kjørt skript for å lage json-filer med data fra mediapipe og openpose
02.03.20	9	Jobbet med å lese json-filene inn i en jupyter-notebook og strukturere dataen
03.03.20	9	Jobbet videre med å strukturere json-filene
04.03.20	8	jobbet med å lage skript for å generere json-filene
05.03.20	8	Jobbet med å lage maskinlæringsmodell.
09.03.20	9	Jobbet med å justere modellen og øke nøyaktighet.
10.03.20	9	Justerte modellen, økte nøyaktigheten til 98
23.03.20	9	Jobbet med å justere modellen, testen den på tilfeldig data
24.03.20	10	Lagde større datasett
25.03.20	9	videre justering av modellen og arbeidet med den nye dataen
26.03.20	10	Lagde større datasett og justerte modellen en del, endre optimizer
27.03.20	10	Jobbet med å bearbeide ny data og justere modellen
30.03.20	9	endret måten vi leste inn data på og jobbet litt med avvikende data.
31.03.20	9	lagde enkelt bokstavers data bedre.
01.04.20	8	forbedret modellen og fikset tensorflow 2 til å kjøre på gpu
02.04.20	9	jobbet med å få til t
03.04.20	8	jobbet med å få bra testing acc.
14.04.20	9	jobbet med lagring av modell og realtime predictions

15.04.20	9	jobbet med å få til 2d array som input/ LSTM.
16.04.20	9	Fortsatte å jobbe med RNN-modellen med et GRU-lag.
17.04.20	10	jobbet med RNN, møte med veileder, melding til to bedrifter om SDK.
20.04.20	8	Jobbet med å få openpose til å kjøre raskere
21.04.20	10	Fylte inn stikkord og underoverskrifter på rapporten og jobbet med RNN-modellen
22.04.20	9	Forbedret RNN-modellen og endre måten vi leste inn data på
23.04.20	9	Testet med mange ulike verdier i den nye CNN-modellen og skrev på rapporten
24.04.20	10	Lagde en TCN-modell og fikk opp mot 93% acc.
27.04.20	9	Skrev på rapporten og jobbet litt med TCN-modellen
28.04.20	8	Jobbet med rapporten
29.04.20	8	Jobbet med rapporten
30.04.20	8	Jobbet med rapporten
01.05.20	8	Jobbet med rapporten
04.05.20	8	Jobbet med rapporten
05.05.20	9	jobbet med rapporten og endret litt på koden
06.05.20	9	jobbet med rapport og strukturering av prosjektet
07.05.20	10	jobbet med rapporten
08.05.20	10	jobbet med rapporten
09.05.20	9	Jobbet med rapporten
10.05.20	9	Jobbet med rapporten
11.05.20	9	Jobbet med rapporten
12.05.20	8	Jobbet med rapporten
13.05.20	10	jobbet med rapporten
14.05.20	11	Leste gjennom rapport
15.05.20	9	Jobbet med ferdigstilling av rapport
16.05.20	9	Ferdingstilling av rapporten
17.05.20	9	Jobbet med kravdokumentasjon og prosjekthåndbok
18.05.20	9	Jobbet med kravdokumentasjon og prosjekthåndbok
19.05.20	9	Ferdigstilte alle dokumentene.
SUM	507	

Logg:

13.01.20

Det første vi gjorde var å forberede en møteinnkalling. Der ble vi enige om sakslista for møteinnkallingen, og senere ble vi enige om hvor og når møte skulle skje. Vi leste også litt om hva som må gjøres og hva slags dokumentasjon som må skrives.

20.01.20

Denne dagen skrev vi på visjonsdokumentet i tillegg til å se litt på hva andre har fått til når det kommer til tegnspråk og maskinlæring. Vi fant flere ulike interessante rapporter og fant ut av mye som vi kan få bruk for.

21.01.20

Idag fikk vi skrevet litt mer under alle punktene i visjonsdokumentet. Vi fant også ut at gjenkjenning av alfabetet er greit å få til, men noen bokstaver kan bli vanskelig, da det er flere som har fått gode resultater på dette. De fleste inkludere ikke de dynamiske bokstavene J og Z, noe som vi skal prøve på. Å få til noe mer enn bare alfabetet blir ganske vanskelig.

29.01.20

Vi ringte en tegnspråklærer og hadde et intervju over telefonen. Av hun fikk vi mye god informasjon om hvilke deler av kroppen som er viktig når det kommer til å forstå tegnspråk.

13.02.20-14.02.20

Sjekket en del på nettet om hvordan tracke hånda og håndbevegelser. Fant mange ulike teknologier, men valgte å prøve mediapipe og openpose fordi mediapipe var laget av google, noe som var lovende og openpose var nevnt av veilederen. Begge virket også veldig profesjonelle og nøyaktig.

17.02.20

Jobbet en del med å installere mediapipe, google sitt program som tracker hender. Hadde litt problemer med å installere opencv. Må reinstallere curl for å få det til å fungere med https-protokollen.

18.02.20

Fikk til å installere og å kjøre mediapipe hand tracking med webkamera på en bærebar pc, etter å ha reinstallert curl og clonet mediapipe-githuben på nytt og installert opencv med vedlagte .sh-filen. Da dette gikk som planlagt fikk jeg kjørt hand tracking programmet. Testet forskjellen på å kjøre mediapipe med CPU og med GPU. Da vi kjørte med GPU fikk vi mye bedre resultater og trackingen virker mye mer nøyaktig og mye raskere. Raske håndbevegelser med få bilder i

sekundet gjør videoen blurry og det blir vanskelig for håndtrackeren å finne hendene. Planlegger å prøve slowmotion-funksjonen på mobilen min.

19.02.20

Jobbet med å installere openpose. Fant ut av at man måtta ha et grafikkort hadde cuda-støtte. Etter å ha lastet ned ulike pakker fikk vi det til å fungere. Bevegelsene virker mer jevne enn Googles mediapipe gjorde, men nøyaktigheten i bevegelser og på ulike tegn var litt dårlig. Fikk også til og hente data ut fra openpose i form av en JSON-fil med x og y-koordinater for hvert punkt, men fikk ikke til å tracke bare hånda. Openpose trenger skulderen og albuen i tillegg.

20.02.20

Lagde treningsdata av hele alfabetet ved å lage videosnutter av hver bokstav i ASL-alfabetet.

26.02.20

Jobbet med å hente x og y posisjonen til punktene med mediapipe. Editerte koden til mediapipe etter å ha funnet noen andre som hadde fått til noe lignende og fikk skrevet punktene til en json-fil. Dette ble gjort ved å editere en av filene, som allerede inneholdt objekter som kunne skrive ut punktene.

27.02.20

Jobbet med å hente ut x- og y-posisjonen fra openpose-håndtrackingen. Dette var enkelt å få til da det bare var å legge til en parameter under kjøringen. Videre jobbet vi også med å strukturere filene i mapper.

28.02.20

Jobbet med å lage json filer fra mediapipe og openpose, ga nytt navn til selvlagd datasett og Kjørte skript for å konvertere videoene våre til lavere oppløsning. Lagde også skript for å hente json-filer fra mediapipe og openpose og lagret filene systematisk i en mappestruktur. Når vi fikk laget skript for å få til dette ble denne prosessen veldig mye enklere, og gjorde det enklere å legge til ny data.

02.03.20

Jobbet med å få lest json-filene inn i en jupyter-notebook. Endret også filnavnet på de genererte mediapipe-filene ved hjelp av skript, slik at navnet lignet på openpose sine genererte filer. I notebooken jobbet vi med å strukturere dataen på en slik måte at den kan brukes i et tensorflow nevralt nettverk. Vi lagde en array med arrays, der hver array innehold en bokstav og en mindre array med alle tilhørende x,y - punkter fra èn fil.

03.03.20

Jobbet videre med å strukturere dataen. Gjorde datasettet om til datatypen Dataset for å kunne bruke den i modellen vår. Fikk til å kjøre modellen, men med ganske dårlige resultater.

04.03.20

Jobbet videre med å strukturere dataen på en smart måte. Prøvde å legge dataen i en pandas dataframe, noe som veldig mange bruker i maskinlæring.

05.03.20

Jobbet med å justere maskinlæringsmodellen, ved å endre parametre som størrelsen på denselagene, aktiveringsfunksjon og optimizers.

09.03.20

Jobbet videre med å justere maskinlæringsmodellen. Fikk litt bedre nøyaktighet, opp mot 77% nøyaktighet.

10.03.20

Jobbet med å forbedre maskinlæringsmodellen ,fikk bedre nøyaktighet, opp mot 98%. Fant en eksempelkode, som hjalp oss med å lage modellen vår, med lag og activation og optimizer. Openpose fikk litt bedre nøyaktighet enn mediapipe.

23.03.20

Jobbet med å teste modellen vår mot andres data. Lagde også metoder for å endre dataen vår som å rotere. På denne måten får vi mere treningsdata slik at modellen vår ikke blir for overfitted. Fant ut av at modellen var overfitted, og at det ikke var realistisk å validere modellen på delen av treningsdataen som den ikke har trent på. Denne dataene er for lik resten av treningsdataen og modellen vil derfor oppnå mye høyere nøyaktighet enn det den burde. Har derfor mye lavere enn 98 % nøyaktighet.

24.03.20

Jobbet videre med å teste modellen, lagde også mere data. Brukte venstre hånd i motsetning til høyre, som vi brukte første gang. Vi må derfor speilvende videoene slik at dataen vår er mer konsistent. Brukte en del tid på å kjøre gjennom den nye data blant annet å kjøre den gjennom mediapipe og openpose.

25.03.20

Jobbet med den nye dataen, prøvde litt å få til med flere arrays som input, men fikk det ikke til. Lagde også nye skript for å speilvende videoene og kjørte dem gjennom mediapipe og openpose-skriptene for å lage json-filer. Justerte modellens variabler for å prøve å få bedre resultat etter at vi endret aktiveringsfunksjonen på de første lagene fra lineær til relu.

26.03.20

Jobbet mye med å justere modellene, ved å endre på tall og optmalisatorerer og andre ting. Jobbet og med å klippe noen videoer fra youtube med hele alfabetet for å få mere treningsdata.

27.03.20

Jobbet med å justere hånden og forbedret noen av videoene fra youtube. Hadde møte med veileder om fremdrift og videre arbeid. Jobbet også med å lage kode for å teste dataen fra youtube for å teset hvordan modellen vår funker på realistisk data.

30.03.20

Jobbet med å forbedre måten vi leste inn data på. Vi endret den slik at hvert startpunkt for hvert frame starter i x,y = (0,0). På denne måten fikk vi bedre resultater med openpose. Vi fikk opp mot 88 % acc på mediapipe og openpose. Men vi har fortsatt problemer med prediction med mediapipe, openpose predicter bedre. Vi fant en måte å fjerne avvikende data fra datasettet med en metode fra scipy som sjekker avstanden til gjennomsnittsverdien. På denne måten fikk vi enda litt bedre med openpose, men fortsatt litt problemer med mediapipe. Opp mot 95 % acc på openpose. Openpose har veldig bra predictions men mediapipe har litt dårligere.

31.03.20

Jobbet med å finne ut hvilke bokstaver modellen vår slet med. Lagde videor til de bokstavene vi slet med på nytt, fordi vi så at openpose ikke klarte å gjenkjenne hånda i mange av videoene. Derfor var dataen på noen av bokstavene dårlig, og vi fikk dårlige resultater på de. Etter å ha laget nye videoer og trent med de fikk vi litt bedre resultater på alle bokstaver. Så også på andre muligheter for å tracke hånda med tanke på å tracke bevegelser bedre.

01.04.20

Jobbet mye med å forbedre modellen, for å få like bra resultat på alle bokstavene. Vi klarer alle bokstavene ganske bra untatt j og z , fordi de har bevegelse og t fordi den en veldig lik. Vi fikk også til å kjøre tensorflow 2 på gpu, slik at treningen av modellen gikk litt raskere.

02.04.20

Jobbet med å finne ut av hvorfor modellen sliter med å gjenkjenne t. Så på hvordan mediapipe og openpose tracket hånda i treningsdataen av t og hvordan den tolket på testdataen. Jobbet og med modellen

03.04.20

Fortsatte og jobbe med det samme som igår.

14.04.20

Jobbet med å finne den beste modellen med openpose, og lagre vektene til en fil. Gjorde dette med en tensorflow-funksjon. Jobbet også med å skrive kode og skript for å kunne kjøre

openpose med predictions i real time. Gjorde dette med et skript som kjørte et python-skript og et shell-skript samtidig. Testet om det funket med en testvideo, men lagde og mulighet for å kjøre det på webkamera.

15.04.20

Jobbet med å få til å sende inn flere arrays samtidig. Fant en måte å gjøre dette på ved hjelp av et LSTM(Long short-term memory) layer i et RNN(Recurrent neural network). Brukt en del til på å justere denne nye modellen for å få bra acc.

16.04.20

Fortsatte å jobbe med å forbedre RNN-modellen. Prøvde et annet type lag som heter GRU. Funket litt bedre.

17.04.20

Jobbet videre med RNN-modellen, forbedret et skript for real time demoen vår og lastet ned en video for å teste den. Hadde møte med veileder. Sende melding to to bedrifter som hadde et hand tracking produkt om vi kan teste det. Planla også hva som skal gjøres den neste ukene.

20.04.20

Jobbet en del med å prøve å få openpose til å kjøre raskere. Hadde noen problemer med opencv og opengl. Brukte en del tid på å builde opencv også. Ble ca. 50% raskere etter at vi la til noen flags ved kjøring av openpose.

21.04.20

Jobbet med å skrive stikkord og underoverskrifter på hovedrapporten. Jobbet også med RNN-modellen, la til flere lag, et GRU og LSTM lag med et SimpleRNN-lag og dropout. Endre også litt på hvordan vi strukturerte dataen.

22.04.20

Jobbet en del med å forbedre RNN-modellen. Endret litt på hvordan vi leste inn dataen. Vi fant også en formel for å normalisere x og y-verdiene med gjennomsnittsverdien og standardavviket for hver frame. Og sjekket om det ga bedre resultater. Ga litt bedre resultater. Vi konstruerte dataen også på en litt annen måte. Vi fant ulike ord og satt de sammen, og for hver bokstav i hele strengen hentet vi ut en tilfeldig sekvens av keypoints for den bokstaven og satte det inn i trenings settet, sammen med y-verdien som er bokstaven. Vi gjorde dette for hele strengen og fikk et større datasett. Prøvde også å lage en CNN-modell og fikk ganske gode resultater på det også.

23.04.20

Testet å kjøre med at learning raten ble mindre over tid, testet også mange ulike verdier for Denselaget og med ulike lengde på hvert array med keypoints, altså hvor mange frames vi trener på samtidig Fikk best resultat på Dense 512 og 30 frames. Fikk også veldig bra på 10 frames så valgte å bruke det sammen med 512 på dense-laget. Begynte også å teste på ulik størrelse på Conv-laget. Overordnet ser det ut som jo større lag vi bruker jo bedre resultater får vi. Skrev også noen avsnitt på rapporten.

24.04.20

Jobbet mye med å å lage en TCN(Temporal convolutional network). Er en type cnn som tydeligvis skal gjøre det bedre en LSTM på oppgaver som LSTM egentlig skal gjøre det best på. Fant dette på en artikkel fra 2018, og fant også en github som hadde et ferdiglaget TCN som vi kunne installere med pip. Fikk opp mot 92% acc med denne nye modellen.

27.04.20

Jobbet en del med rapporten, skrev introduksjon og relevans, akronymer, teori om tegnspråk og teori om maskinlæring. Jobbet også litt med å kjøre modellen.

28.04.20-01-05.20

Jobbet med rapporten. Skrev på teoridelen, og innledning. Kjørte også modellene av og til

04.05.20

Lagde litt mer data på J og Z, prøvde litt forskjellige parametre på modellene. Skrev også mye notater på rapporten.

05.05.20

Skrev på rapporten og strukturerte koden vår litt bedre i ulike notebooks og ulike notebook-celler. Kjørte også flere av modellene og fikklitt bedre resultater enn før.

06.05.20

jobbet med rapport og strukturering av prosjektet

07.05.20

Jobbet med rapporten, møte med veileder.

08.05.20-14.05.20

Jobbet med rapporten og begynte å lese gjennom

15.05.20

Ferdigstilling av rapporten og gjennomlesning.

16.05.20

Ferdigstilling av rapporten

17.05.20

Jobbet med kravdokumentasjon og prosjekthåndbok

18.05.20

Jobbet med kravdokumentasjon og prosjekthåndbok

19.05.20

Ferdigstilte alle dokumenter og hele prosjektet.

Tegnspråk med maskinlæring
Visjonsdokument

Versjon 1.2

Revisjonshistorie

Dato	Versjon	Beskrivelse	Forfatter
21.01.2020	1.0	Skrev dokument	Sigurd og Eivind
05.02.2020	1.1	Mindre justeringer	Sigurd og Eivind
25.02.2020	1.2	La til noen avsnitt	Sigurd og Eivind

Innholdsfortegnelse

1. Innledning

Hensikten med dette dokumentet er å få frem hvilket problem produktet skal løse og hvem som får bruk for produktet. Dokumentet skal også fortelle om produktets rolle, forutsetning og avhengigheter og funksjonelle og ikke funksjonelle krav. I tillegg skal dokumentet inneholde alternativer til produktet vårt og gi en overordnet oversikt over hva produktet skal utføre.

Prosjektet skal foregå over en periode på 5 måneder i samarbeid med 3D Motion Technologies og utføres som et bachelorprosjekt.

2. Sammendrag problem og produkt

2.1 Problemsammendrag

Problem med	Dagens system er at de er ukomplett
Berører	Folk som er døve eller jobber med døve
Som resultat av dette	Ett problem er at systemene krever dyre kameraer
en vellykket løsning vil	være enkel å bruke og gi nøyaktige tilbakemeldinger

2.2 Produktsammendrag

For	3D Motion Technologies
Som	Vil oversette tegnspråk med video
Produktet navngitt	Er et gjenkjenningssystem
Som	Er lett tilgjengelig og enkelt å bruke

I motsetning til	Dagens systemer
Har vårt produkt	Ikke behov for avanserte kameraer og mye prosessorkraft.

3. Overordnet beskrivelse av interessenter og brukere

3.1 Oppsummering interessenter

Navn	Utdypende beskrivelse	Rolle under utviklingen
Døve og stumme	Vil kunne bli forstått selv om motparten ikke kan tegnspråk.	Kan hjelpe med forståelse av tegnspråk og datainnsamling.

3.2 Oppsummering brukere

Navn	Utdypende beskrivelse	Rolle under utviklingen	Representert av
De som omgås døve eller stumme	Den hyppigste brukeren av systemet.	En person som kan brukerteste systemet, for å oppfylle kravene. Kan også verifisere oversetningen.	Seg selv

3.3 Brukermiljøet

Målet er at systemet skal passe inn i flere ulike miljøer. Eksempler på dette kan være omsorgssenter og opplæringssenter. Systemet må derfor være lett å bruke og passe godt inn i slike miljøer.

Målet er også at systemet skal passe inn i et hverdagslig miljø. Med dette menes at det skal kunne tas opp en video med tegnspråk med for eksempel en mobiltelefon og deretter kunne oversette tegnspråket. Produktet bør også passe inn i et undervisningsmiljø, da planen er at det også skal bli brukt i undervisning.

3.4 Sammendrag av brukernes behov

Behov	Prioritet	Påvirker	Dagens løsning	Foreslått løsning

Oversetting	Høy	Systemets brukelighet	Tolk eller å lære tegnspråk selv.	System som oversetter tegnspråk via video
Vise tegnspråk på avatar	Lav	Systemets helhet	Finnes flere løsninger, f.eks. Simax [1]	Visualisere en allerede filmet video med tegnspråk

3.5 Alternativer til vårt produkt

- Simax oversetter fra tekst til tegnspråk og animere tegnspråket på en avatar. Animasjonen er en del av prosjektet vårt, men er ikke hovedfokuset til oppgaven.
- Xsens er et firma som har gode alternativer til motion capture. [2]

4. Produktoversikt

4.1 Produktets rolle i brukermiljøet

Produktets rolle vil være å oversette for de som ikke kan tegnspråk. Det vil være et hjelpemiddel for å lettere kommunisere med døve eller stumme, og for å brukes i sammenheng med opplæring.

4.2 Forutsetninger og avhengigheter

- Fungerende programvare(Tensorflow, Blender)
- Regelmessige møter med veileder
- Alle parter har samme mål og forventninger med prosjektet. En endring i mål eller forventninger fører til endring av dette dokumentet.
- Regelmessig oppmøte på alle medlemmer i gruppa.

5. Produktets funksjonelle egenskaper

- Fremstille punkter på kroppen ved hjelp av video/bilde
- Bruke disse punktene til å oversette tegnspråk
- Kunne oversette staving på tegnspråk.

[1] https://simax.media/?lang=en

[2] https://www.xsens.com/motion-capture

6. Ikke-funksjonelle egenskaper og andre krav

- FURPS+
- Nvidia-skjermkort med Quda support
- Python
- Tensorflow
-

7. Referanser

- [1] Simax lesedato 25.02.20 https://simax.media/?lang=en
- [2] Xsens motion capture lesedato 25.02.20 https://www.xsens.com/motion-capture

Prosjektnr. 64

Tegnspråk
Systemdokumentasjon

Versjon 1.0

Revisjonshistorie

Dato	Versjon	Beskrivelse	Forfatter
15.05.20	1.0	Dokumentet ble ferdigstilt	Sigurd Løite Jacobsen og Eivind Alfsvåg Johansen

Innholdsfortegnelse

Innledning

Dette dokumentet er skrevet i forbindelse med prosjektet Tegnspråk med Maskinlæring. Hensikten med dokumentet er å dokumentere systemet. Dokumentet inneholder en detaljert beskrivelse av fil- og katalogstrukturen for prosjektet, der også skriptfilers bruksområde forklares. Hvordan systemet installeres og kjøres beskrives også. Her beskrives også systemets avhengigheter og programvarebibliotek som brukes

Prosjektstruktur

Mappestrukturen er modellert på slutten av dokumentet. Under modelleres prosjektets mappestruktur ved hjelp av “tree”-kommandoen i Linux. I den første mappen jupyter ligger kildekoden, samt de lagrede filene som trengs for å laste inn hver maskinlæringsmodell. Kildekoden er lagret i form av .ipynb-filer for hver enkelt modell som er filtypen for en jupyter notebook-fil. I tillegg ligger maskinlæringsmodellene for openpose lagret i mappen Saved_ModelOpen og Saved_ModelOpen Backup. Videre i mappen Openpose_RealTime lagres JSON-filene som brukes i realtime-demoen og i test_data ligger all generert data. Eksempel video inneholder et eksempel på hvordan videosnuttene så ut. json_files_mediapipe og json_files_openpose inneholder begge mapper med navn tilsvarende ASL-alfabetet der hver av disse mappene inneholder JSON-filer generert av enten Openpose eller Mediapipe tilhørende hver bokstav. Scripts-mappen inneholder skriptfiler brukt til behandling av filene, blant annet for sletting, flytting, generering av data og behandling av videofiler. Speilvend-mappen er en mappe som ble brukt for videofiler som skulle speilvendes ved hjelp av et skript. I mappen New_vids inne i Temp_videos la vi videofilene som det skulle genereres data fra slik at vi ikke trengte å generere data fra alle de tidligere videofilene på nytt hver gang. I mappen Test_videos, strukturert på samme måte som json_files_mediapipe og json_files_openpose, ble det lagt videosnutter som det skulle lages test- og valideringsdata fra. Disse videosnuttene er videoen hentet fra nett. Til slutt i videos ble de selvlagde videosnuttene lagt.

Installasjon og kjøring

Merk at Linux 19.04 ble brukt i dette prosjektet, men installering av avhengighetene under bør fungere like bra på tidligere versjoner.

Avhengigheter:

- Linux(https://ubuntu.com/tutorials/tutorial-install-ubuntu-desktop#1-overview)
- Python 3.7 (https://www.python.org/downloads/)
- Jupyter notebook(https://jupyter.readthedocs.io/en/latest/install.html)
- Openpose (https://github.com/CMU-Perceptual-Computing-Lab/openpose/blob/master/doc/installation.md)
- Mediapipe (https://github.com/google/mediapipe/blob/master/mediapipe/docs/install.md)
- Tensorflow 2.1.0-rc0(https://www.tensorflow.org/install)
- TCN laget (https://github.com/philipperemy/keras-tcn)
- Numpy 1.18.3 (https://pypi.org/project/numpy/)
- Scipy 1.2.2 (https://www.scipy.org/install.html)
- Matplotlib 3.0.2 (https://matplotlib.org/users/installing.html)
- Seaborn 0.10.0 (https://seaborn.pydata.org/installing.html)
- sklearn 0.22.2 (https://scikit-learn.org/stable/install.html)

Installasjon

1. Last ned Tegnspraak.zip og pakk ut innholdet.
2. Installasjonen av openpose skal legges inn i openpose mappen i Tegnspraak mappen, og installeres som beskrevet på Openposes github.
3. Mediapipe skal installeres direkte i Tegnspraak mappen og installeres som beskrevet på Mediapipes github.
4. Installer resten av avhengighetene som beskrevet over.

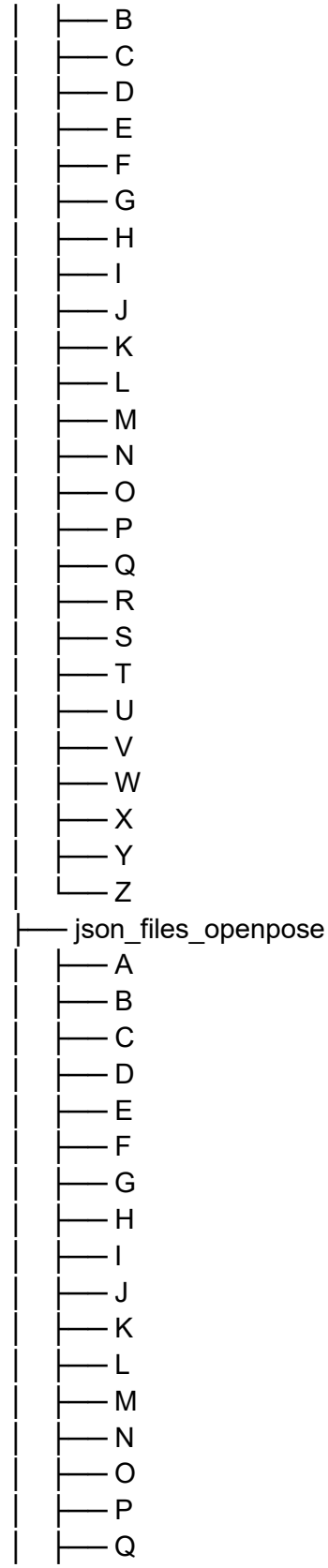
For å få best mulig resultat vil det være nødvendig å installere Openpose og Mediapipe slik at de kan kjøre på gpu.

Kjøring av demo:

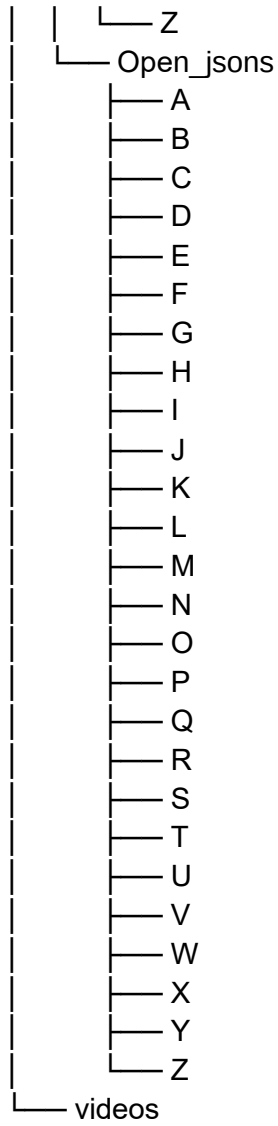
1. Åpne en terminal i mappen Tegnspråk
2. Skriv kommandoen ./RealTimeVideo.sh {video.mp4} eller ./RealtimeWebcam.sh
3. Det skal skal starte openpose og det vil samtidig skrives ut hvilket tegn modellen tror det er.

Mappestruktur:
Tegnspraak

```
├── jupyter
│   ├── ModelMedia
│   ├── ModelOpen
│   │   ├── ANN
│   │   ├── CNN
│   │   ├── RNN
│   │   └── TCN
│   ├── Saved_ModelMedia
│   ├── Saved_ModelOpen
│   │   ├── 93.42TCN
│   │   │   ├── assets
│   │   │   └── variables
│   │   ├── 93.915ANN
│   │   │   ├── assets
│   │   │   └── variables
│   │   ├── 93RNN
│   │   │   ├── assets
│   │   │   └── variables
│   │   ├── 95CNN
│   │   │   ├── assets
│   │   │   └── variables
│   │   └── 98CNN
│   │       ├── assets
│   │       └── variables
│   └── Saved_ModelOpen Backup
│       ├── 93.42TCN
│       │   ├── assets
│       │   └── variables
│       ├── 93.915ANN
│       │   ├── assets
│       │   └── variables
│       ├── 93RNN
│       │   ├── assets
│       │   └── variables
│       └── 95CNN
│           ├── assets
│           └── variables
├── openpose
├── Openpose_RealTime
└── test_data
    ├── Eksempel video
    ├── json_files_mediapipe
    │   ├── A
```

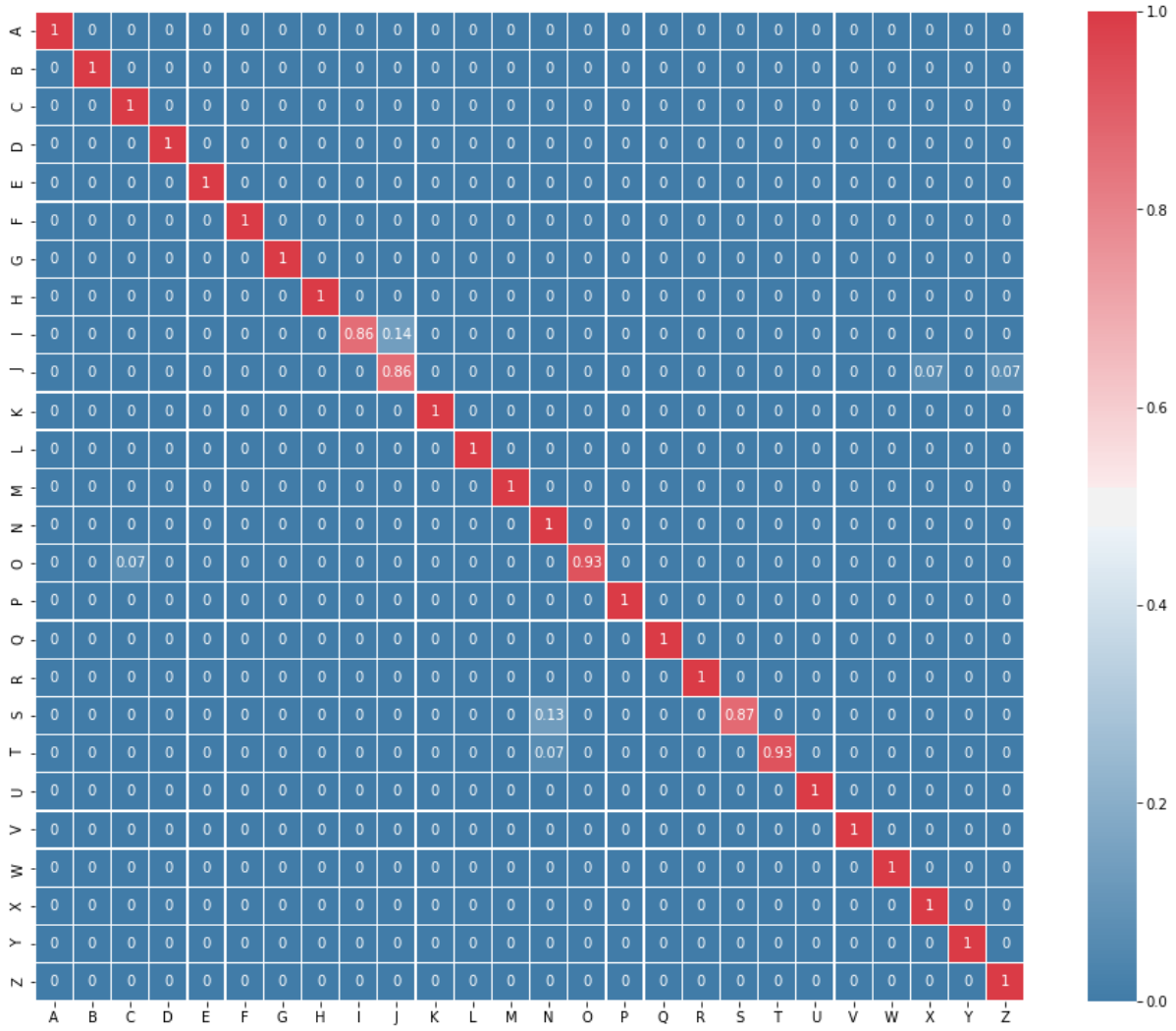

B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z
json_files_openpose
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q

```
│   ├── R
│   ├── S
│   ├── T
│   ├── U
│   ├── V
│   ├── W
│   ├── X
│   ├── Y
│   └── Z
├── scripts
│   └── old
├── Speilvend
│   ├── Vids_Input
│   └── Vids_Output
├── Temp_videos
│   └── New_Vids
├── Test_videos
│   ├── Media_jsons
│   │   ├── A
│   │   ├── B
│   │   ├── C
│   │   ├── D
│   │   ├── E
│   │   ├── F
│   │   ├── G
│   │   ├── H
│   │   ├── I
│   │   ├── J
│   │   ├── K
│   │   ├── L
│   │   ├── M
│   │   ├── N
│   │   ├── O
│   │   ├── P
│   │   ├── Q
│   │   ├── R
│   │   ├── S
│   │   ├── T
│   │   ├── U
│   │   ├── V
│   │   ├── W
│   │   ├── X
│   │   ├── Y
```

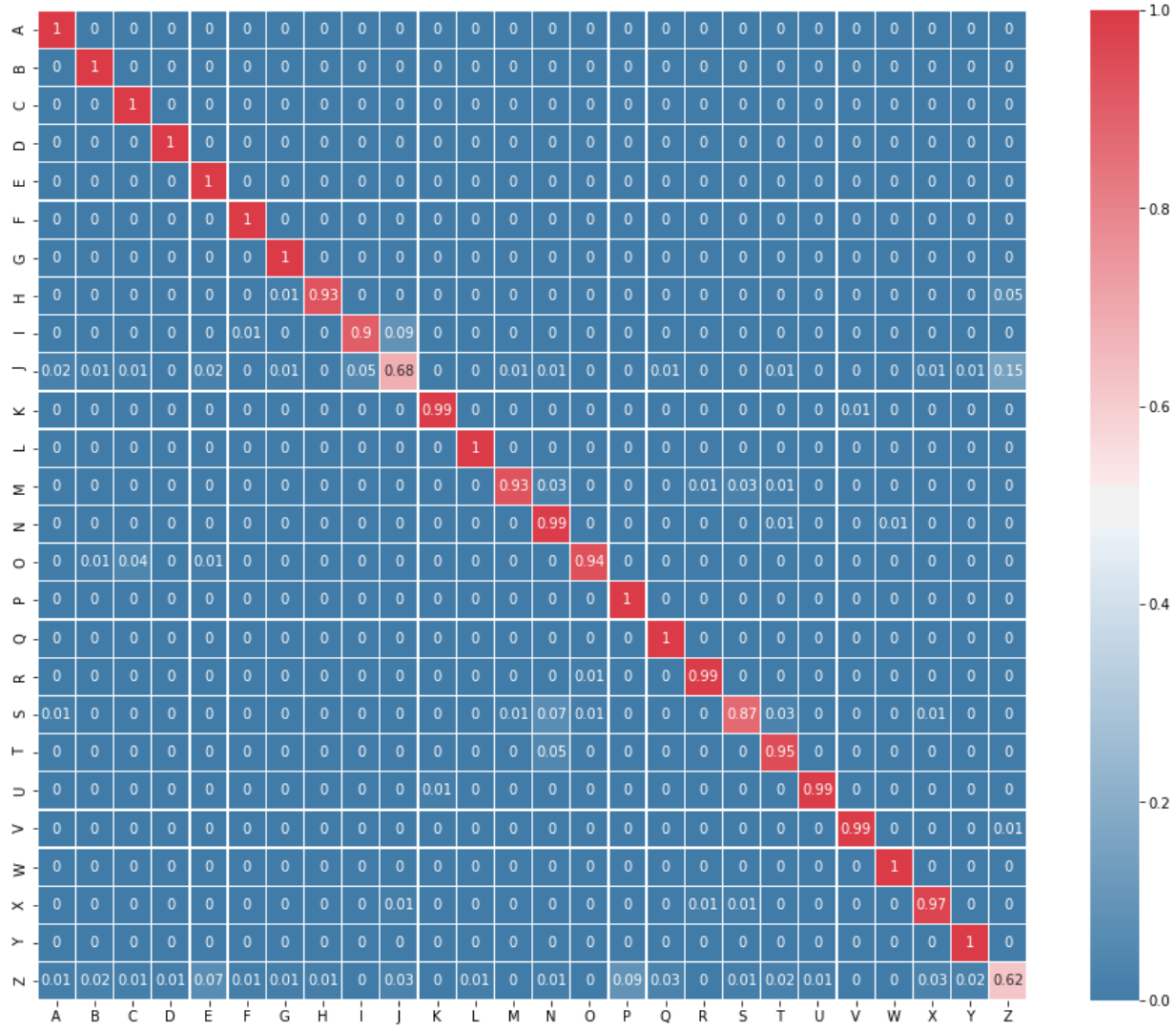

Z
Open_jsons
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z
videos

Heatmaps

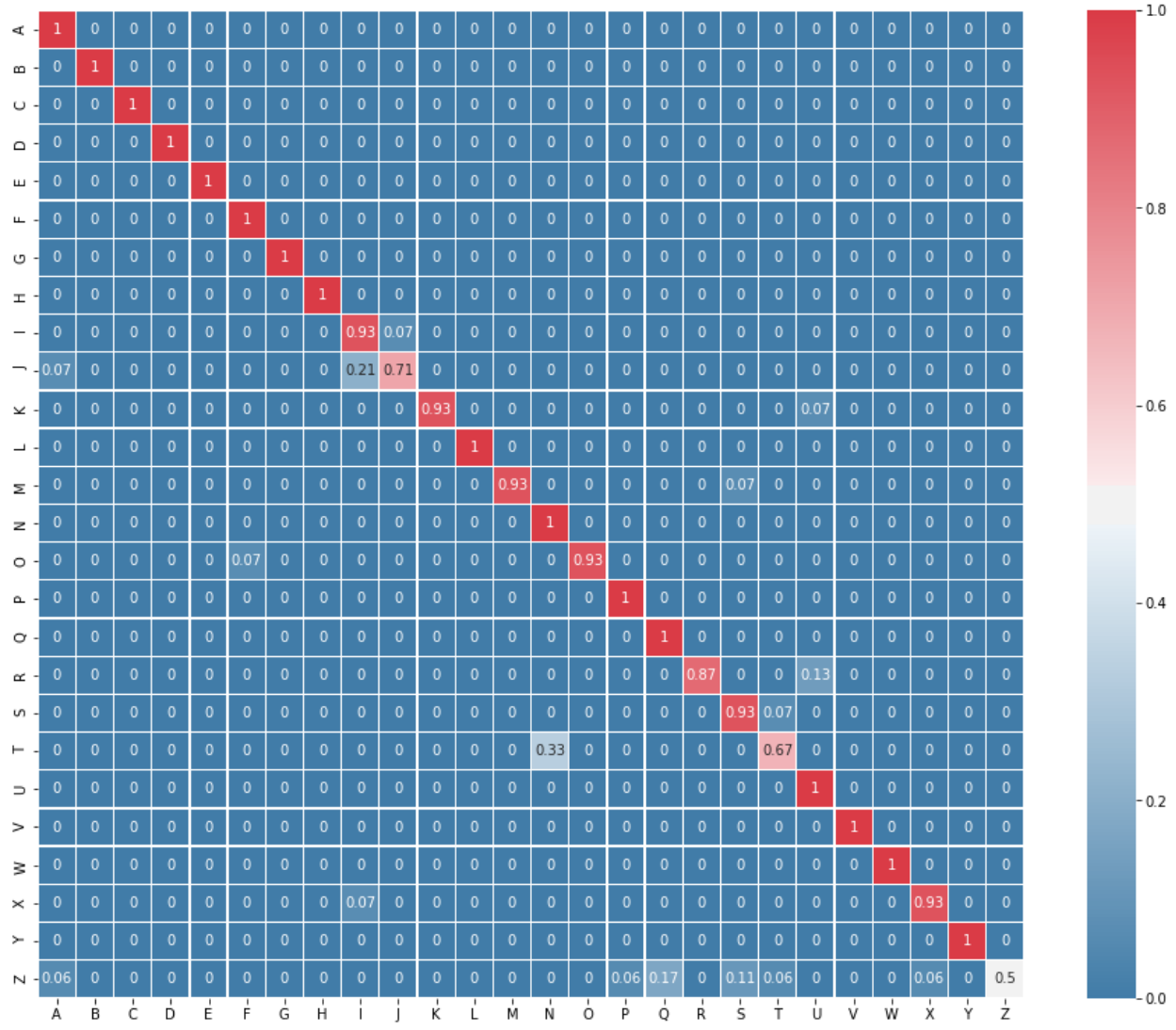
CNN



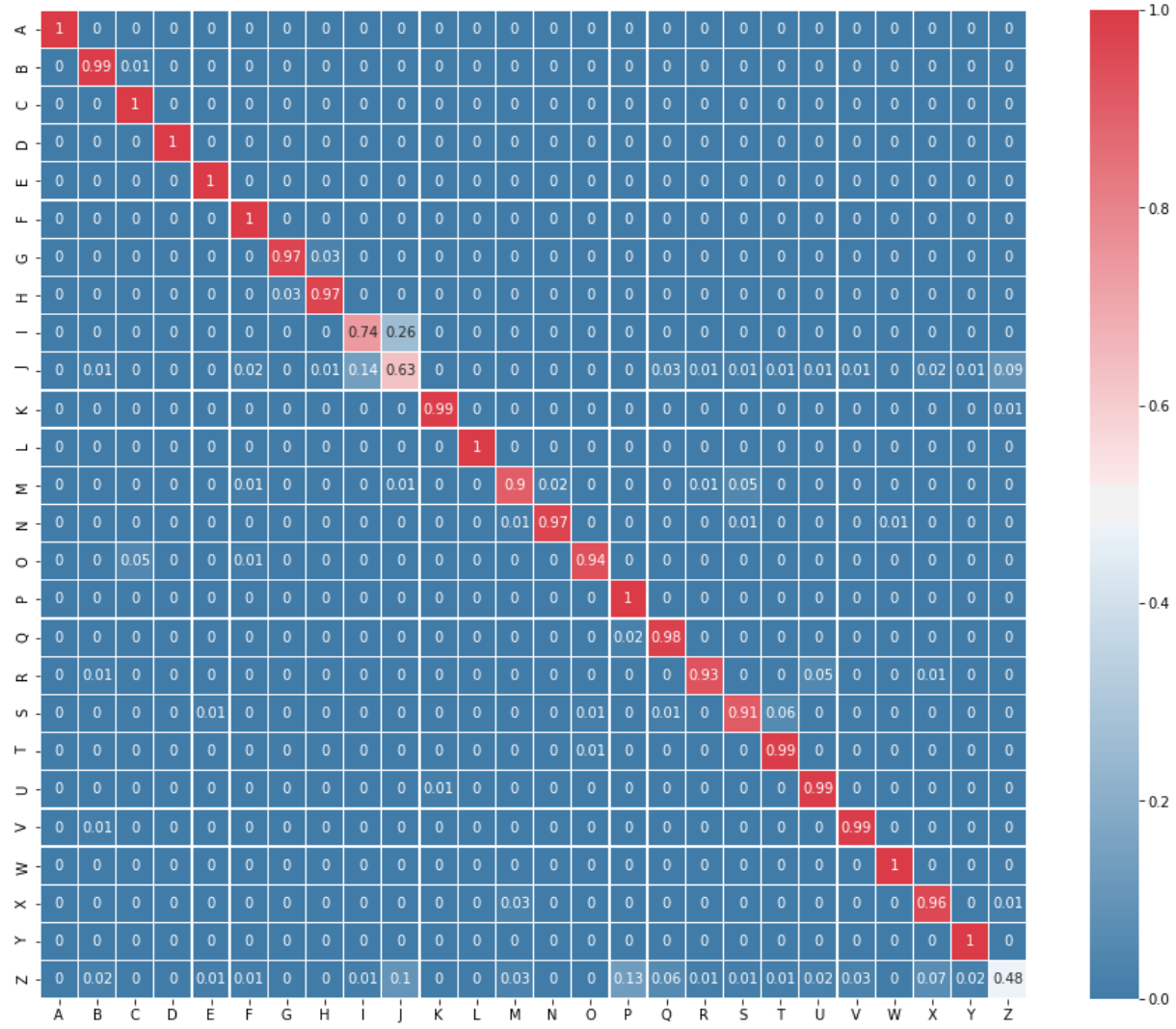
ANN


A 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
B 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
C 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
D 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
E 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
F 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
G 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
H 0 0 0 0 0 0 0.01 0.93 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0.05
I 0 0 0 0 0 0.01 0 0 0.9 0.09 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
J 0.02 0.01 0.01 0 0.02 0 0.01 0 0.05 0.68 0 0 0.01 0.01 0 0 0.01 0 0 0.01 0 0 0 0.01 0.01 0.15
K 0 0 0 0 0 0 0 0 0 0 0.99 0 0 0 0 0 0 0 0 0 0 0.01 0 0 0 0
L 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
M 0 0 0 0 0 0 0 0 0 0 0 0 0.93 0.03 0 0 0 0.01 0.03 0.01 0 0 0 0 0 0
N 0 0 0 0 0 0 0 0 0 0 0 0 0 0.99 0 0 0 0 0 0.01 0 0 0.01 0 0 0
O 0 0.01 0.04 0 0.01 0 0 0 0 0 0 0 0 0 0.94 0 0 0 0 0 0 0 0 0 0 0
P 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
Q 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
R 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0.01 0 0 0.99 0 0 0 0 0 0 0 0
S 0.01 0 0 0 0 0 0 0 0 0 0 0 0.01 0.07 0.01 0 0 0 0.87 0.03 0 0 0 0.01 0 0
T 0 0 0 0 0 0 0 0 0 0 0 0 0 0.05 0 0 0 0 0 0.95 0 0 0 0 0 0
U 0 0 0 0 0 0 0 0 0 0 0.01 0 0 0 0 0 0 0 0 0 0.99 0 0 0 0 0
V 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0.99 0 0 0 0.01
W 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
X 0 0 0 0 0 0 0 0 0 0.01 0 0 0 0 0 0 0 0.01 0.01 0 0 0 0 0.97 0 0
Y 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
Z 0.01 0.02 0.01 0.01 0.07 0.01 0.01 0.01 0 0.03 0 0.01 0 0.01 0 0.09 0.03 0 0.01 0.02 0.01 0 0 0.03 0.02 0.62
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
1.0
0.8
0.6
0.4
0.2
0.0

RNN


A 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
B 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
C 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
D 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
E 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
F 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
G 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
H 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
I 0 0 0 0 0 0 0 0 0.93 0.07 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
J 0.07 0 0 0 0 0 0 0 0.21 0.71 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
K 0 0 0 0 0 0 0 0 0 0 0.93 0 0 0 0 0 0 0 0 0 0.07 0 0 0 0 0
L 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
M 0 0 0 0 0 0 0 0 0 0 0 0 0.93 0 0 0 0 0 0.07 0 0 0 0 0 0 0
N 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
O 0 0 0 0 0 0.07 0 0 0 0 0 0 0 0 0.93 0 0 0 0 0 0 0 0 0 0 0
P 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
Q 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
R 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0.87 0 0 0.13 0 0 0 0 0
S 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0.93 0.07 0 0 0 0 0 0
T 0 0 0 0 0 0 0 0 0 0 0 0 0 0.33 0 0 0 0 0 0.67 0 0 0 0 0 0
U 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
V 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
W 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
X 0 0 0 0 0 0 0 0 0.07 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0.93 0 0
Y 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
Z 0.06 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0.06 0.17 0 0.11 0.06 0 0 0 0.06 0 0.5
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
1.0
0.8
0.6
0.4
0.2
0.0

TCN


A 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
B 0 0.99 0.01 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
C 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
D 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
E 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
F 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
G 0 0 0 0 0 0 0.97 0.03 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
H 0 0 0 0 0 0 0.03 0.97 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
I 0 0 0 0 0 0 0 0 0.74 0.26 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
J 0 0.01 0 0 0 0.02 0 0.01 0.14 0.63 0 0 0 0 0 0 0.03 0.01 0.01 0.01 0.01 0.01 0 0.02 0.01 0.09
K 0 0 0 0 0 0 0 0 0 0 0.99 0 0 0 0 0 0 0 0 0 0 0 0 0 0.01
L 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
M 0 0 0 0 0 0.01 0 0 0 0.01 0 0 0.9 0.02 0 0 0 0.01 0.05 0 0 0 0 0 0 0
N 0 0 0 0 0 0 0 0 0 0 0 0 0.01 0.97 0 0 0 0 0.01 0 0 0 0.01 0 0 0
O 0 0 0.05 0 0 0.01 0 0 0 0 0 0 0 0 0.94 0 0 0 0 0 0 0 0 0 0 0
P 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
Q 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0.02 0.98 0 0 0 0 0 0 0 0 0
R 0 0.01 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0.93 0 0 0.05 0 0 0.01 0 0
S 0 0 0 0 0.01 0 0 0 0 0 0 0 0 0 0.01 0 0.01 0 0.91 0.06 0 0 0 0 0 0
T 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0.01 0 0 0 0 0.99 0 0 0 0 0 0
U 0 0 0 0 0 0 0 0 0 0 0.01 0 0 0 0 0 0 0 0 0 0.99 0 0 0 0 0
V 0 0.01 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0.99 0 0 0 0
W 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
X 0 0 0 0 0 0 0 0 0 0 0 0 0.03 0 0 0 0 0 0 0 0 0 0 0.96 0 0.01
Y 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
Z 0 0.02 0 0 0.01 0.01 0 0 0.01 0.1 0 0 0.03 0 0 0.13 0.06 0.01 0.01 0.01 0.02 0.03 0 0.07 0.02 0.48
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
1.0
0.8
0.6
0.4
0.2
0.0

Intervju med Anne Mallene Wassmo

Hvor viktig er de forskjellige kroppsdelene når det gjelder å bli forstått når det gjelder tegnspråk?
Alt er viktig fra håndflate rotasjon til finger posisjon, øyenbryn, munn, fingre, håndflate, fjes munn, øyenbryn
Hvor viktig er dybdesyn for å forstå tegnspråk?
Ikke så viktig
Er ansikts grimase viktig for å bli forstått?
Ansiktsgrimase er veldig viktig. Ett ord kan tolkes på mange måter bare med forskjellige ansiktsgrimaser
Er det noen tegn eller lignende som gjør slik at du flytter på deg tar skritt fram eller lignende hopper?
nei
Hvordan er grammatikken til tegnspråk i forhold til norsk?
Det er forskjellige dialekter i Norge som gjør at tegnspråksord har forskjellige måter å vises på.
Grammatikk er bygd opp annerledes (tid sted jeg) så: (jeg var på skolen i dag) blir (I dag skole jeg)
Har du en noen inputt å komme med fra din erfaring med tegnspråk som du tror kan hjelpe oss eller noe du tror vi burde vite.
Veldig forskjell på person til person, noen som bruker lite munn, lite mimikk
Tegnspråk er veldig lite standardisert

Ble anbefalt ting så se videre på:
Bacheloroppgave for tegnspråk
Temsok ordstilling, orale komponenter, tegn rommet, lager tegning av setningen/historie(blir historien)
For å se hva Anne Malene Wassmo mener med mye mimikk se på:
Nrk døve gudstjeneste
Poesi på tegnspråk