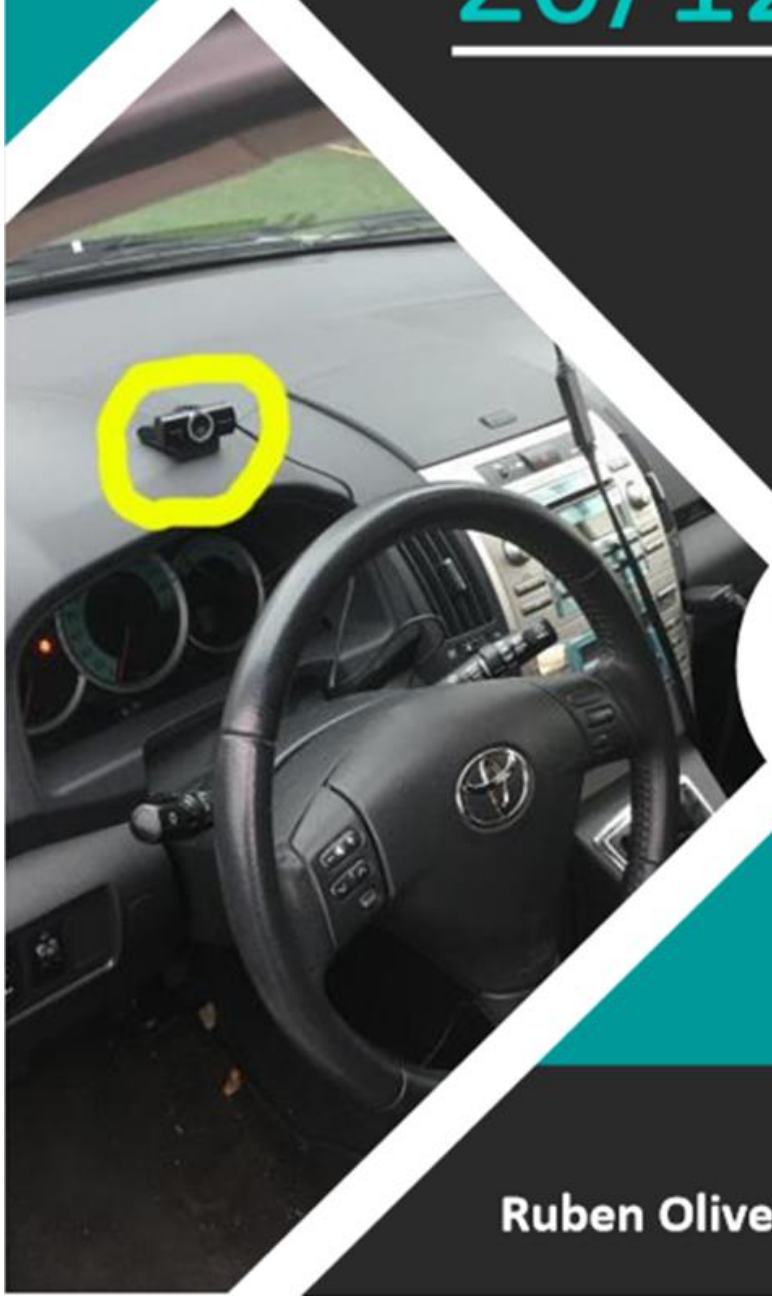
20/12-2020



Deep Learning
Træthed
Løsning

Ruben Oliver Jonsman

1 Resumé

Dette projekt omhandler problemstilling, at folk sætter sig bag et rat i en bil, når de er påvirket af træthed og medvirker i en trafikulykke. I første del af projektet, kigges der på de samfundsmæssige problematikker problemet har ved sig, samt hvilke problematikker produkter der har til formål at løse problemet kan have. Der analyseres problemet for at finde årsager og virkninger, samt dokumentation for at problemet er et problem der er behov for at løse. Den anden del af projektet omhandler Deep Learning og dets principper, for at skabe en forståelse om hvordan det kan anvendes til problemløsning. Principper såsom, loss funktioner, vægte, kunstige neuroner, gradient descent, back og forward propagation dækkes også. Derudover præsenteres der også hvordan man konverterer billeder om til data og behandler dem ved brug af Deep Learning. I tredje del af projektet analyseres og diskuteres løsningen på problemet, samt redegøres for, hvordan produktet løser problemet. Der diskuteres hvilke problematikker der var under implementering af produktet i brugssituationer, samt hvordan produktet skal blive implementeret i samfundet. Til sidst, diskuteres, vurderes og perspektiveres der, hvordan Deep Learning og dets principper kan blive anvendt i andre aspekter af hverdagen, såsom, lufthavne og dets sikkerhed, sundhedssystemet og automatisering af det, sprogteknologi og hvordan man kan nedlægge sprogbarrieren. Derudover også hvordan det k an hjælpe mennesker. I konklusionen konkluderes der at prototypen vil kunne løse problemstillingen, hvis det implementeres i samfundet.

2 Indhold

3 Indledning, problemstilling og problemformulering

Offentligt transport bliver brugt mere end aldrig før, hvilket betyder at offentlig transport chauffører har lange arbejdsdage. Disse mennesker kører i busser og tog fra tidligt om morgenen til sent om aftenen. Der er også mennesker som går på arbejde tidligt og kommer sent hjem. Menneskerne sætter sig ud i deres biler sent om aftenen, hvor de er trætte efter en lang arbejdsdag. De vil gerne hurtigt hjem, og her kan de derfor komme til at køre for hurtigt samt være søvnige og uopmærksomme på selve kørslen. Derudover er det største problem lastbilchauffører som sidder i en lastbil op til 7-8 timer i døgnet. De mennesker der gør ovenstående, øger chancen for trafikuheld i dagligdagen.
Derfor undrer jeg mig over, at der ikke er nogen præventiv løsning til mindskning af uheld, der er opstået pga. mangel på opmærksomhed og søvnighed. Teknologien er tilstrækkelig til at lave applikationer og fysiske produkter der kan detektere søvnighed og uopmærksomhed. Hvorfor implementeres det ikke og hvordan kan jeg finde en løsning til problemet? For at finde viden om emnet vil jeg besvare nogle analyserende, vurderende og redegørende spørgsmål:

- Jeg vil redegøre for, hvad Deep Learning er?
- Jeg vil analysere, hvordan og hvorfor søvnighed påvirker samfundet?
- Jeg vil vurdere, hvordan man kan bruge Deep Learning til at mindske uheld forsaget af søvnighed og uopmærksomhed?
- Jeg vil derudover også diskutere og perspektivere, hvorledes Deep Learning kan anvendes med fordele/ulemper i et overvågningssamfund?
- Derudover vil jeg demonstrere, hvordan man kan anvende Deep Learning i en IT-løsning.

Ovenstående spørgsmål kan kobles ned til: **Hvordan påvirker søvnighed samfundet, og hvordan kan man bruge Deep Learning til at mindske uheld forsaget af søvnighed?**

Jeg vil drage viden fra en rapport, ”Trafikulykker for året 2018” af Vejdirektoratet samt kigge på udviklingen i trafikulykker i en rapport fra Vejdirektoratet. Derudover vil jeg lave samfundsanalyser og produktanalyser ud fra teknologi fagets metoder. Derudover vil jeg gøre brug af Deep Learning, en under-gren af Machine Learning indenfor programmering

---

4 Metode

I opgaven arbejdes der med fagene Teknologi og Programmering. Metoder fra programmering og teknologi sammen for at skabe en helhedsforståelse af problem, prototype og kode. Programmering er et anvendeligt og videnskabeligt fag, hvilket leder til kreation af applikationer. Således arbejdes der med metoder som Deep learning og Loss Funktioner, for at beskrive hvordan prototypen fungerer. Teknologi er også et anvendeligt og videnskabeligt fag, her leder det til analyse af samfund og analyse af produkt. Der er blevet lavet samfundsanalyser for at etablere om der er et problem, samt produktanalyse for at optimere og belyse produktet fra flere perspektiver.

Til opgavens redegørende del benyttes videnskabelige artikler om emnet, faglitteratur, samt data og undersøgelser, hvor der med kritisk sans vurderes, hvad der er brugbart og repræsentativt for området der undersøges.

Til programmering er der blevet brugt Artificiel Neural Network (ANN) med Deep Learning for at lave ansigtsgenkendelse. Artificiel Neural Networks bliver brugt til at lave ansigtsgenkendelse, da der bruges data til at øge effektiviteten, hvilket et ANN kan behandle hurtigere end andre metoder. Der er også blevet valgt ikke at gå i dybden med matematikken bag Artificiel Neutral Networks, da det ikke er formålet med opgaven.
I Teknologi bruges der metoder som, problemanalyse, produktanalyse og teknologivurdering. Problemanalysen laves for at skabe et grundlag for om der reelt er et problem. Der er blevet lavet produktanalyse for at vide hvor produktets styrker og svagheder er, med formålet om at ændre de svage punkter. Derudover er der lavet teknologivurdering for at vurdere, hvordan teknologien skal implementeres i samfundet. Derudover er der også blevet brugt iterativ udvikling for at optimere på prototypen, efter brugertest. Jeg har valgt ikke at gå ind i virksomhedsaspektet af teknologifaget, da det ikke vil være relevant for en opgave skrevet i programmering og teknologi, som har stort fokus på produktet. En problematik i opgaven, var at lave ansigtsgenkendelse uden at gemme følsomme data. Metoderne fra programmering og teknologi kombineres i diskussion og perspektivering, hvorved metoderne samles og perspektiveres til andre aspekter af samfundets hverdag.

5 Samfundsmæssigt

I dette kapitel vil de samfundsmæssige problemstillinger, blive analyseret.

5.1 Problemanalyse

Problem: *Mennesker sætter sig bag et rat, imens de er påvirket af søvnighed og medvirker i en trafikulykke.*

5.1.1 Hvad er træthed?

Træthed er i normal sammenhæng defineret som en følelse af mangel på energi. (Kessing, 2018) (Nationnal Libray of Medicine National Institutes of Health, 2013) (Ukendt, Den danske ordbog - Moderne dansk sprog, ukendt)

Der er mange måder hvorpå man kan opdage om et menneske er træt. Man kan kigge på electroencephalogram, electrocardiogram, photoplethysmogram (Cosic, 2008). Dog er disse besværlige at måle da man skal måle hjerneaktiviteten, så derfor kan man gøre brug af nogle lidt mere anvendelige metoder, såsom at observere om:

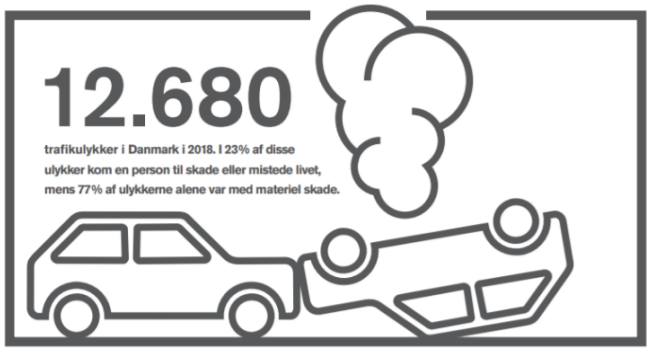
- Man gaber
- Ens øjenlåg bliver tunge
- Man kan ikke huske hvad der skete et par minutter siden.
- Du har svært ved at se hvad der står på skilte og på papir.

Rådet for Sikker Trafik vurderer også at det er disse punkter der er vigtigst at holde øje med når man kører bil.

Derudover har Karolinska Instituttet i Stockholm, Sverige, lavet en undersøgelse om hvornår og om man kan se om en person er i søvnmangel. De kom frem til at mennesker der var i søvnunderskud havde, hængende øjenlåg, rødere øjne, hævede øjne, mørkere cirkler under øjnene, blegere hud, hængende mundvigene. (Nationnal Libray of Medicine National Institutes of Health, 2013)

5.1.2 Hvor stort er problemet?

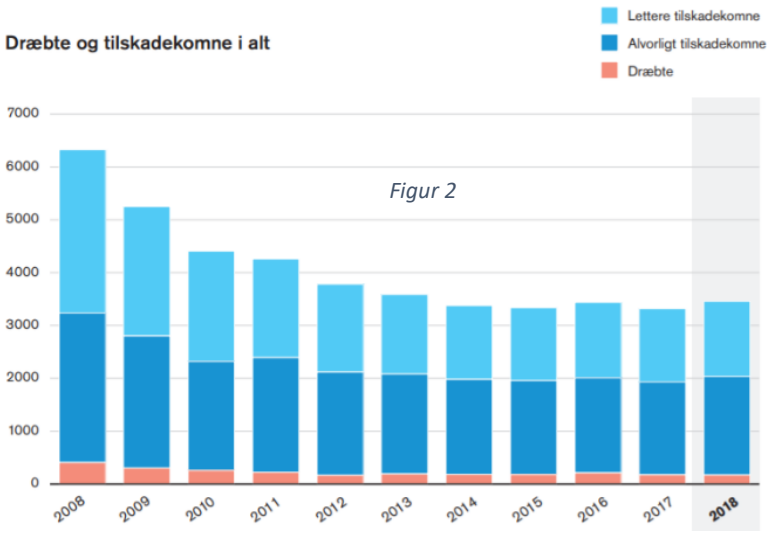
De sundhedsskadelige og samfundsskadelige konsekvenser grundet ulykker er store. Dette ses specielt i en af Vejdirektoratets rapporter om netop, bilulykker. 1.425 bilister og passagerer kom lettere til skade, 1.862 kom alvorligt til skade og 171 mennesker mistede livet i 2018 grundet en trafikulykke. Derudover var der 921 dræbte og tilskadekomne cyklister, hvilket er det højeste antal siden 2008. Dog ses der også en udvikling, hvor 65 af de 171 der mistede livet var bilister, dette er det laveste antal i 10 år. Der ses også at 17 af de dræbte var mellem 18-24 år, hvilket igen er det laveste i mere end 10 år. (Hvid I. &., Maj 2019)



Figur 1

5.1.3 Er problemet aktuelt? Er problemet voksende?

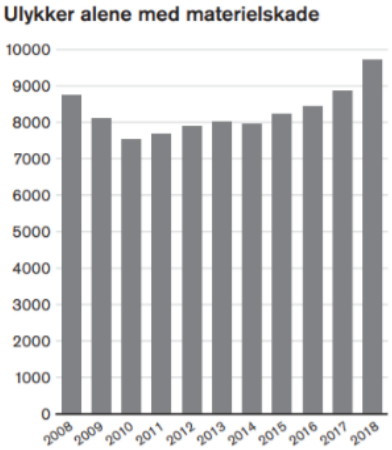
Danmarks veje er ifølge statistikken blevet mere sikre, idet der er færre ulykker om året. Der var færre der mistede livet i 2018 i trafikken, dog var der flere der kom til skade i trafikken. Dette kan være grundet mere sikkerhed i biler, og måske sundhedsvæsnets evner. (Hvid I. &., Ukendt) (Hvid I. &., Maj 2019)



Figur 2

Underligt nok, er det bevist at der er flere ulykker der medvirker materielskade på biler og andre genstande end der tidligere har været. Dette tyder på at der reelt ikke er færre ulykker, men bedre sikkerhed. Der ses i figur 4 at i 2008 at der var omkring 9000 ulykker med materiel skade, i 2013, 8000 og i 2018, 9500 ulykker med materiel skade.

Figur 4



Figur 3

5.1.4 Hvordan vil problemet se ud om ti år?

Ifølge den her udvikling som ses på figur 2, kan det antages at der ikke vil forekomme færre ulykker med personskade. Der kan blot antages, dog med brug af regressions analyse, at det kan blive approksimeret/gættet til hvad det vil være om 1 og 10 år. Dette vil være meget upræcist, da der ikke er specielt meget data, samt at der er mange faktorer der ikke tages højde for. Ved brug af regressionsanalyse kan det antages at der vil være et meget lille fald i antal personskade ulykker i de kommende år, hvis samme udvikling som der har været de sidste 5 år gentages. Der skal ske en radikal ændring ellers vil antal ulykker ikke falde meget mere.

5.1.5 Hvem omfatter problemet?

Problemet omfatter alle i samfundet, lige meget om en person transporterer sig på ben, i bil, på cykel, vil personen være i risiko for at medvirke i en ulykke forsaget af en uopmærksom eller træt bilist. Man vil kunne gå hjem efter skole i et meget sikkert miljø og område, men en bilist ville kunne dræbe fodgængeren, ved blot at være træt og uopmærksom.

5.1.6 Hvor mange ulykker skyldes træthed?

Træthed eller at sove forklarer en lang række ulykkesfaktorer, såsom: (HVU, 2015)

- Manglende bevidsthed
- Manglende orientering
- Utilstrækkelig orientering
- Forkert placering
- Hastighed i forhold til forholdende, hastighedsgrænsen og manøvren
- Fejltolkning og vurdering
- Forkert manøvre eller reaktion.

Havarikommissionen skriver at de har analyseret 12 ulykker af typen mødeulykke. De er kommet frem til, at ulykkesfaktoren er manglende bevidsthed i 6 ud af de 12 mødeulykker. Den manglende bevidsthed skyldes i 4 tilfælde træthed eller søvn. I ét tilfælde skyldes det et ildebefindende og i ét tilfælde skyldes det en blanding af alkohol og narkotika. Dette er ikke et repræsentativt datasæt, dog belyser det situationen lidt.

Ved forkert placering, har havarikommissionen analyseret at træthed optræder i 6 ulykker og forklarer at i 5 af tilfældene, at føreren var faldet i søvn og derfor ikke ved bevidsthed. I sidste ulykke er det grundet manglende orientering grundet træthed. Disse bilister ignorerer tegnene på træthed og kører videre. Bilisterne opfatter ikke kørsel med en træt fører som risikabelt for en selv og omverdenen. (HVU, 2015)

Det er svært at bevise eller konkludere at en ulykke var grundet træthed, da der i de fleste tilfælde ikke er nogen klar indikation på at føreren er medvirkede i ulykken fordi personen var træt. I nogle specialtilfælde, kan der ses at sporene fra bilens dæk, er meget ujævne og snoede, hvilket kan være et tegn på at føreren er træt.

5.1.7 Hvad er årsagerne og virkningerne?

Årsagerne og virkningerne er med til at etablere og skabe et overblik over hvad der kan løses. Der kan vælges at fokusere på at løse stress i hverdagen for at mindske problemet. Der kan også blive fokuseret på at løse de psykiske problematikker som problemet skaber. Der kan også blive valgt at løse problemet midlertidigt.

Årsag til problemet:

- Stress i hverdagen
- For lange arbejdsdage
- Mangel på søvn
- Langt til og fra arbejde
- Jernmangel (Brandt, 23.07.2003)
- Sygdomme
- Depression

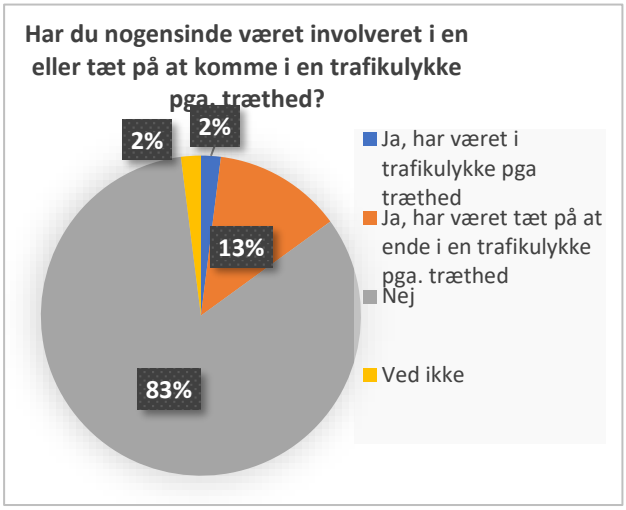
Virkninger af problem/problemet medfører:

- Flere ulykker'
- Forældreløse børn
- Følelsesmæssige problemer
- Psykiske problemer
- Fysiske problemer
- Søvnforstyrrelser, PTSD
- Udgifter til diverse, statstilskud til helbred m.m.

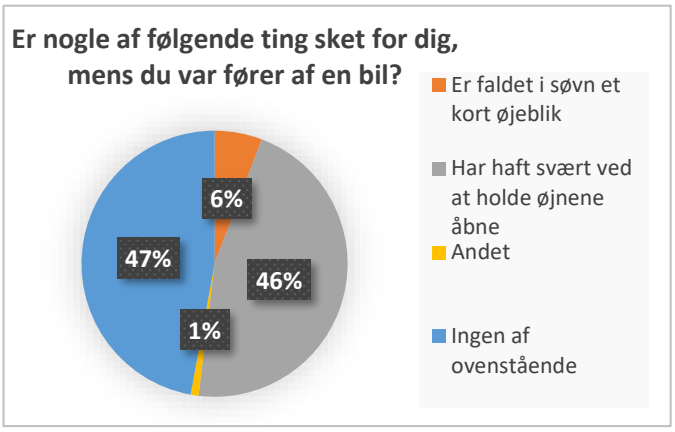
5.1.8 Hvordan ser den danske befolkning problemet?

I en undersøgelse om træthed og trafikulykker lavet af Kantar Gallup for Gjensidige Forsikring, ses der befolkningens syn på problemet samt om de har oplevet det. Undersøgelsen er lavet på baggrund af 1.560 repræsentativt udvalgte danskere.

Der ses i rapporten og undersøgelsen, at størstedelen af de repræsentative danskere ikke har været ude for en trafikulykke grundet træthed. Dog har 46% af de udvalgte danskerne, haft svært ved at holde øjnene åbne, hvilket kan medføre trafikuheld. Dette er et foruroligende højt tal.



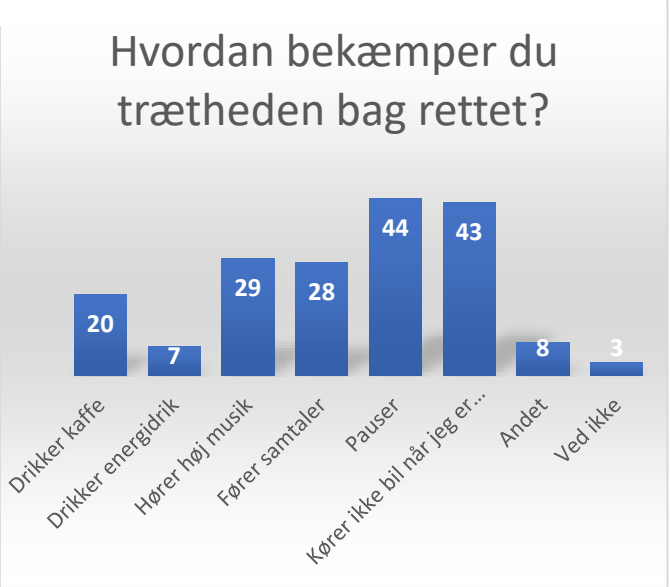
Figur 5



Figur 6

5.1.9 Hvad sker der hvis problemet løses?

Hvis problemet løses, vil det i teorien mindske antallet af ulykker markant. Hvilket vil betyde at Danmark vil bruge færre penge på at hjælpe de tilskadekomne. Dette betyder at Danmark som helhed vil have flere penge der kan bruges på andre sektorer, som uddannelse, sikkerhed, mere velfærd m.m.



Figur 7

5.2 General Data Protection Regulation– data bliver ikke gemt

GDPR gælder for alle virksomheder og organisationer, som sælger varer og ydelser og opbevarer personlige data om borgere i Europa, inklusive virksomheder fra andre kontinenter. Den giver borgere i EU og EØS større kontrol over deres personlige data og sikkerhed for, at deres information beskyttes i hele Europa. (EU, 2018)

Ifølge GDPR defineres personlige data som enhver form for information relateret til en person. Dette kunne være navn, billeder, e-mailadresse, bankoplysninger, opslag på sociale medier, oplysninger om lokation, helbredsinformation og IP-adresser. Alle disse oplysninger eller information kan i de forkerte hænder blive misbrugt. (EU, 2018)

Gdpr.dk har opstillet nogle generelle regler for GDPR, hvor enhver virksomhed skal:

- Føre en fortegnelse (”kortlægning af virksomhedens processer, hvori der behandles personoplysninger. Der er særlige krav til denne fortegnelse. Alle der behandler personoplysninger skal ifølge loven have lavet denne fortegnelse.”)
- Dokumentere at lovgivningens principper for god databehandling efterleves.
    - Principper indeholder, Ansvarlighed, formålsbegrænsnings, dataminimering, lovlighed, rimelighed og gennemsigtighed, rigtighed, integritet og fortrolighed, opbevaringsbegrænsning.
- Dokumentere at virksomheden har indført passende tekniske og organisatoriske foranstaltninger.
- Oplyse kunder og ansatte om hvordan deres data behandles.
- Bevise, at virksomheden efterlever lovgivningen fx hvis der anvendes samtykke, databehandlere, mv.

6 Deep Learning

6.1 Hvad er Deep learning

For at kunne definere, hvad Deep Learning er, er man nødt til at vide, hvordan det opstod og hvad der ligger til grund for det. Deep Learning er en under-gren til Machine Learning, og for at forstå Machine Learning skal man gå ned til det basale og forstå koncepterne ved Artificiel Intelligence (AI) også kendt som kunstig intelligens. AI kort opsummeret er, en computer der simulerer intelligent adfærd. Der er mange definitioner af en AI, dette er blot en af mange og den jeg vil referere til (Merriam-Webster, 2019).

Kigges der meget forsimplet på en AI, kan den beskrives som en masse såkaldte ”if-else statements”. Dette er grundet at en AI bliver programmeret til at opføre sig på en bestemt måde, i en bestemt situation. Har man en AI der er programmet til at spille skak, og man så sætter den til at spille spillet ”Go” i stedet, vil computeren ikke have nogen mulig måde at vinde på. Her kan der ses at AI’en kun er funktionel i den ene situation den er lavet til at blive benyttet i.

6.2 Machine Learning

Nu da Artificiel Intelligence er blevet beskrevet, kan Machine Learning og Deep Learning blive dækket. Machine Learning er kort beskrevet processen hvorpå en compter er i stand til at forbedre effektiviteten ved at kontinuert at inkorporere ny data ind i en allerede eksisterende model. (Inc., 2019)

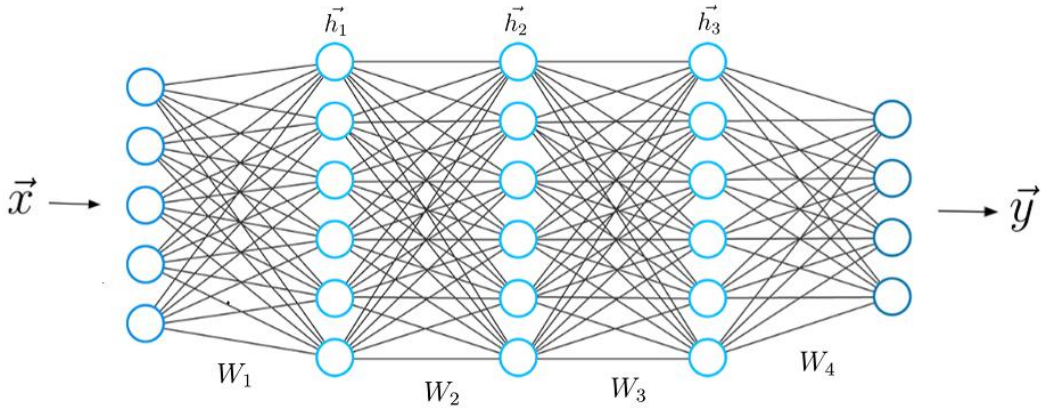
Machine Learning inkorporerer ”klassiske” algoritmer for mange forskellige opgaver såsom, clustering, regression, og klassifikation. Machine Learning algoritmer skal trænes på data, jo mere data algoritmer får, jo bedre bliver dens prediction.

Machine Learning leder op til en stor række forskellige automatiserede opgaver, IT-virus søgning og vejr forudsigelse er blot et af de mange opgaver. Ved IT-virus opdagelse, giver man et program testdata fra mange forskellige IT-systemer der har haft virus og siger at sådan ser et IT-system ud der har virus. Derefter tjekker programmet ens data fra et nyt IT-system, og konkluderer om det har virus ud fra en række kriterier. Samme fremgangsmåde er der med vejr forudsigelse, der gives testdata, og derefter kan ens program forudsige om det vil regne m.m, ud fra en række kriterier der gives til programmet.

Et eksempel på en Machine Learning algoritme kunne være, Perceptron. Denne algoritme bygger på koncepterne bag menneskets hjerne. Denne algoritme gør brug af kunstige neuroner, som beskrives i næste afsnit. (Sharma, 2017)

6.3 Deep learning

Deep Learning er en under-gren indenfor Machine Learning med metoder baseret på hovedsageligt kunstige neurale netværk (ANN). Årsagen til at Deep Learning er en under-gren til Machine Learning, er grundet behovet for data til at lære at løse opgaver. Learning kan være supervised, semi-supervised eller unsupervised. Deep learning gør brug af mange lag til gradvist at finde sammenhænge og funktioner fra rå-data eller input. Eksempelvis kunne det være billede behandling. Eftersom der kan blive brugt ANN i et Machine Learning program som i Perceptron algoritmen, er der stadig forskelle mellem Machine Learning og Deep Learning. Selvom Perceptron er bygget op på en ANN struktur, er den meget ineffektiv i forhold til et Deep Learning ANN system. Selvom et ANN er blevet implementeret i et Machine Learning program, vil et ANN Deep Learning system, kunne løse opgaver som Machine Learning aldrig ville kunne løse.

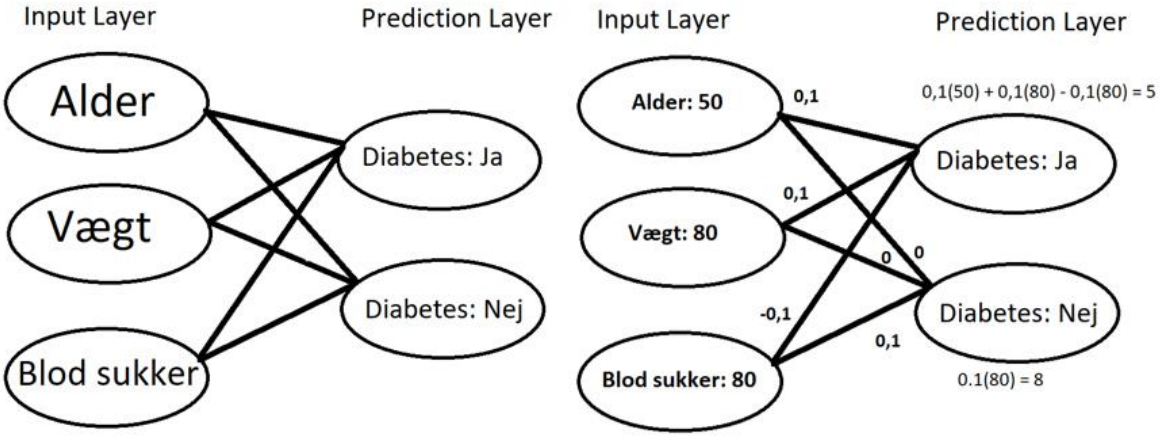


Figur 8 (Oppermann, 2019)

På figur 8 er der afbilledet et ANN system. I et ANN system er der et input og et output. Imellem input og output er der mange faser inputtet går igennem før det konverteres til noget brugbart. Der ses også en masse cirkler og linjer der forbinder cirklerne. Disse cirkler repræsenterer en masse kunstige neuroner. Hvert neuron kan sende et signal til et andet neuron. Det neuron der modtager signalet kan derefter bearbejde den viden den nu har fået givet, og derefter følge samme procedure som det tidligere neuron har foretaget sig, og til sidst vil alle neuronerne have fået et samlet output som brugeren derefter kan bruge. (Hansen, 2019)

6.4 Vægte

Vægte bruges til at forbinde hvert neuron i et lag til det næste lag. Vægte bestemmer hvor stærk forbindelsen mellem neuroner er. De vægte der ses på figur 8 som er defineret med et stort W, er dem der er med til at definere hvad outputtet bliver. Ens dataværdi vil blive ganget med vægten, hvorefter alle værdier bliver lagt sammen hvilket også ses på figur 9. (Hansen, 2019)



Figur 9

Den opnåede værdi er indikationen af predictionen. Den laveste værdi vil være det programmet forudsiger er svaret. Dermed, hvis vægtene ændrer sig, ændrer ens prediction også. Eftersom disse vægte er så vigtige, er det i programmørens interesse at have så præcise vægte som muligt. Der er 3 måder at præcisere vægte på, som spiller sammen: ”loss funktion”, ”gradient descent” og ”back propagation”.

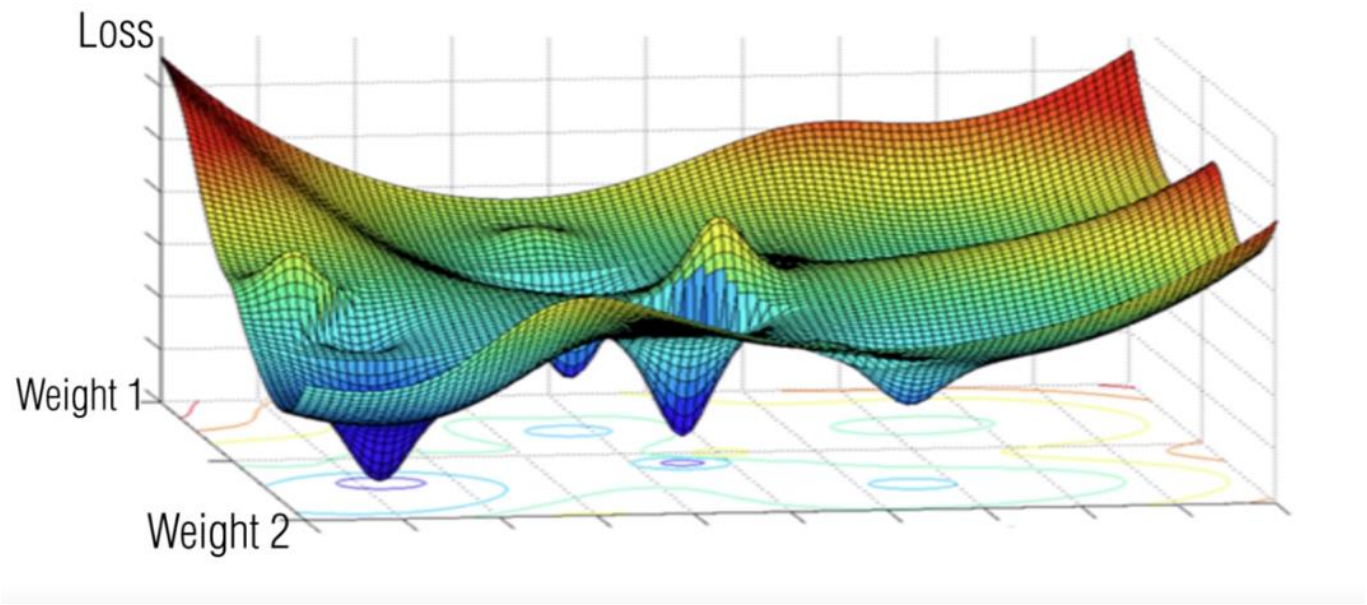
6.4.1.1 Loss funktion

En "loss funktion" måler hvor god en models predictions er. For at en loss funktion kan bruges, er der nogle argumenter der skal tjekkes af, disse er: ens prediction og den korrekte værdi man prøver at forudsige. I Python kan det skrives på denne måde: (Hansen, 2019) (Becker, 2018)

```
Loss = f(actual, predicted)
```

Som en konvention ved brug af en loss funktion, er lave værdier de bedste. Det betyder at hvis ens predictede værdi er tæt på den korrekte værdi, vil loss funktionen returnere en lille værdi. Er ens predictede værdi, langt fra den korrekte værdi, vil loss funktionen returnere en høj værdi.

Et givent datasæt, hvor modellen enten er præcis eller upræcis vil afhænge af modellens vægte. Den numeriske værdi af loss funktionen på et givent datasæt vil også ændre sig hvis vægtene ændres. Vægtene påvirker vores prediction, og dermed også loss funktionens værdi. For at forstå dette kigges der på et eksempel. I figur 10, har vi to vægte hvilket betyder at modellen kan blive plottet i en 3D-graf. I praksis er dette urealistisk, da der aldrig vil være en brugssituation med kun to vægte. Grunden til at der bruges to i dette tilfælde, er fordi en model med flere vægte ikke vil være i stand til at blive plottet i en graf. I grafen har vi Loss Funktionen op ad Z-aksen og vægte på de andre akser. Målet er at finde de laveste punkter i grafen for at finde de optimale vægte. (Becker, 2018)



Figur 10
(Becker, 2018)

Den måde hvorpå man finder de laveste punkter er besynderlig. Det kan beskrives som, hvis man står i en ørken i tusmørke, man kan ikke se noget og man går i blinde. Man får til opgave at finde de laveste punkter i den her ørken. Man vil føle sig frem til hvilken retning der går nedad og hvilken retning der går opad. Man vil tage et skridt i retningen af den stejleste retning, hvorefter man vil føle sig frem igen. Dette forsættes indtil der ikke er lavere at gå. Dette er måden hvorpå, "gradient descent" fungerer. (Becker, 2018)
I Gradient Descent kigges der på data, og man ser hvilke vægte man skal ændre for at få en lidt lavere loss funktion værdi, og vægten ændres dermed i den retning, hvis man har fået en lavere loss funktion værdi. Dette fortsættes indtil man har en tilfredsstillende værdi.

Når der laves Gradient Descent, bruges der normalt ikke al ens data, da dette ville betyde at man skulle gennemgå umenneskelige store mængder udregninger. Derfor kigges der på brødstykker af datasættet.

Disse brødstykker bliver kaldt for en ”batch”. En Batch er et antal rækker i et tabellarisk datasæt der bliver brugt til at udregne en loss funktion værdi. Bruges der billeder som data er batchen, antallet af billeder der bliver brugt til at udregne loss funktion værdien, hvilket fortsættes indtil man har brugt al data. Hver gang man har kørt igennem sit data, kaldes for en ”epoch”. Trinvist forbedrer vægtene sig for hver ”epoch”, hvilket betyder at hvert billede eller data punkt, vil blive brugt til at forbedre vægtene mere end en gang. (Becker, 2018)

6.5 Forward og back propagation

I forward propagation, kommer informationen ind i input laget, hvorefter det propagerer fremad igennem netværket af kunstige neuroner, hvor vi til sidst vil finde vores output. Værdierne vil blive sammenlignet med det forventede resultat.

Back propagation er en af de vigtigste byggesten i et ANN. Ifølge en artikel, ”Learning representations by back-propagating errors” i 1989 af Rumelhart, Hinton and Williams. Definerer de back propagation som:

“Repeatedly adjust the weights of the connections in the network so as to minimisze a measure of the difference between the actual output vector of the net and the desired output vector.” (David E. Rumelhart, 1986)

“The ability to create useful new features distinguish back-propagation from earlier, simpler methods… “ (David E. Rumelhart, 1986)

Med andre ord, back propagation sigter efter at minimere loss funktionens værdi ved at justere vægte i ANN (se afsnit 6.3). Justeringsniveauet bliver bestemt ud fra værdien af loss funktionen.

6.6 Deep learning på billeder

Når der skal implementeres Deep Learning på en række billeder, er billederne nød til at blive manipuleret for at man kan gøre brug af dem. På figur 11 ses det håndskrevne tal 2. Tallet består af pixels, hvor nogle pixels er fuldstændig sorte, nogle er fuldstændig hvide og nogle har forskellige nuancer af grå. Pixels kan anses som stående i rækker og i kolonner, derfor vil det være naturligt at konvertere billedet til en matrice. Konverteres dette 2 tal til en matrice ville det se ud som på figur 12. Hver værdi repræsenterer intensiteten af farven i den givne pixel. Top-venstre hjørne i matricen vil repræsentere det top-venstre hjørne af billedet. Dette er tilfældet når man har sort-hvid billeder. I farvebilleder er der en ekstra dimension. For hver pixel der bliver gemt i en matrice, er der en anden, en tredje og en fjerde matrice der viser hvor, rød, grøn og blå den givne pixel er. Dermed er matricen en sammenkædning af 3 matricer. (Becker, 2018)

Figur 11

0	0	200	150	0	0
0	143	55	99	222	0
0	188	0	0	181	0
0	0	0	200	0	0
0	0	149	0	0	0
0	245	202	140	225	0
0	0	0	0	0	0

Figur 12

7 Produkt

I dette kapitel vil produktet blive beskrevet, analyseret, diskuteres.

7.1 Produktbeskrivelse

Produktet går ud på at finde ud af ved brug af matematiske formler, hvornår en person er træt. Dette skal implementeres i en bil, for at mindske trafikulykker forsaget af træthed. Opfanger produktet at føreren af bilen er træt eller uopmærksom, vil den afspille en høj lyd for at vække føreren.

Produktet er skrevet i programmeringssproget Python, ved brug af følgende moduler med forskellige funktioner:

- Dlib - Til at lave ansigtsgenkendelse
- Scipy - Til at udregne "Euclidean Distance" mellem to 1-D vektorer.
- Opencv2 - Til at visualisere processen
- Numpy - Til at bruge matematiske funktioner
- Sound - Til at spille lyde
- Imutils - Gør visualisering nemmere

7.2 Krav til produkt

Ved at opstille krav til et produkt gøres det tydeligt, hvad man skal ligge fokus på, samt hvad man forventer produktet skal kunne. Hvis produktet ikke opfylder de opsatte krav, skal produktet forbedres eller kasseres.

Jeg har opsat følgende krav til produktet, baseret på min problemanalyse:

- Skal kunne detektere om en person er træt eller ej.
- Skal ikke gemme data / Skal stemme overens med GDPR
- Skal være funktionelt og præcis
- Skal være sikkert og ikke være forstyrrende
- Skal starte ved opstart af bil
- Skal ikke kunne slukkes manuelt

7.3 Produktanalyse

7.3.1 Hvorfor gør programmet det det gør.

Det vil være simpelt at implementere Karolinskas resultater (se kapitel 5.) i et program, da der blot skal sammenlignes farver. Dog vil det kræve at man gemmer billeder af brugeren, hvilket kan stride imod GDPR. Dette er årsagen til at disse løsninger ikke er blevet valgt. Hvis denne løsning skulle bruges, ville alt data være nødt til at blive krypteret, og gå igennem flere former for databeskyttelse.

Det vil også være simpelt at implementere om en person gaber og om deres øjne lukkes, falder i og bliver tunge, da det blot er afstanden mellem topøjenvippe og bundøjenvippe.

7.3.2 GDPR og program

Eftersom produktet der præsenteres i denne opgave, gør brug af Deep Learning og dermed data, er det vigtigt at GDPR ikke overtrædes. Derfor vil produktet ikke gemme data på nogen måde. Der vil ikke blive gemt indsamlet data efter brug af produktet. Dataet vil blive brugt til at finpudse vægte for at få et mere korrekt resultat, dog når programmet lukkes, vil disse data ikke være tilgængelige. Derudover vil vægte også blive nulstillet, hvilket betyder at der ikke kan blive ekstraheret viden ud fra vægtene. Alt dette

betyder at programmet vil køre langsomt kort efter opstart, da den er nødt til at indsamle data under processen.

7.3.3 Virkemåde

I følgende afsnit beskrives der, hvordan programmet finder ud af om personen er træt.

Figur 13

Ved brug af landmarks i ansigtet (figur 13) kan der kodes at man vil finde afstanden mellem 2 punkter. Dette er især vigtigt, da det er den måde hvorpå jeg finder ud af om en person er træt.

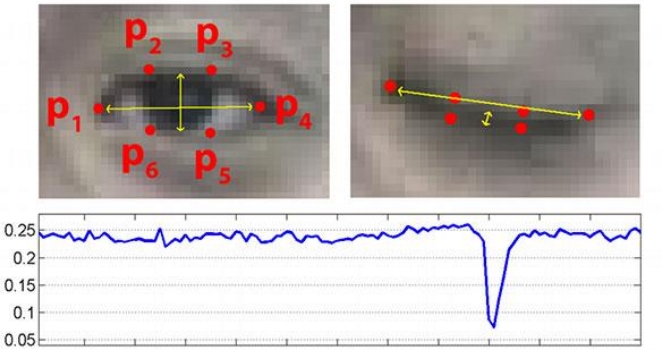
Programmet finder ud af om en person er træt, ved at kigge på øjnene og munden.

7.3.3.1 Øjnene

Programmet finder afstanden mellem to punkter i øjet, p1-p4, p2-p6, p3-p5, hvilket ses på figur 14. Dette gør den ved at finde det givne punkt, og konvertere det til et array. Eksempel på et array man kan få: [[191 137], [194 131], [200 129], [207 129], [202 134], [197 136]]. Dette betyder, [[p1], [p2], [p3], [p4], [p5], [p6]]

Der vil nu blive udregnet "euclidean" afstand mellem punkterne. Derefter er det blot at vide, at hvis afstanden er mindre end ens grænseværdi, er personen træt.

På figur 14 billede ses der hvordan afstanden bliver opfanget.

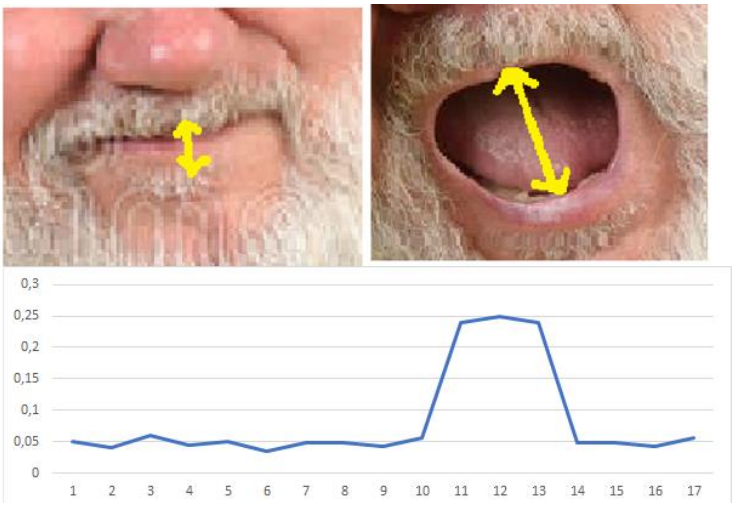


Figur 14
Top-venstre: En visualisering af et øje der er åbent.
Top-højre: En visualisering af et øde der er lukket.
Bunden: Visualisering af "eye aspect ratio" over tid.
Dykket i grafen repræsenterer et blink.
https://www.pyimagesearch.com/2017/05/08/drowsiness-detection-opencv/

Øjet fungerer som en kontakt til lys. Når øjet er åbent, sker der ikke noget. Når øjet lukkes, klikkes kontakten ind, og lyden spilles. Åbnes øjet igen slippes kontakten.

7.3.3.2 Munden

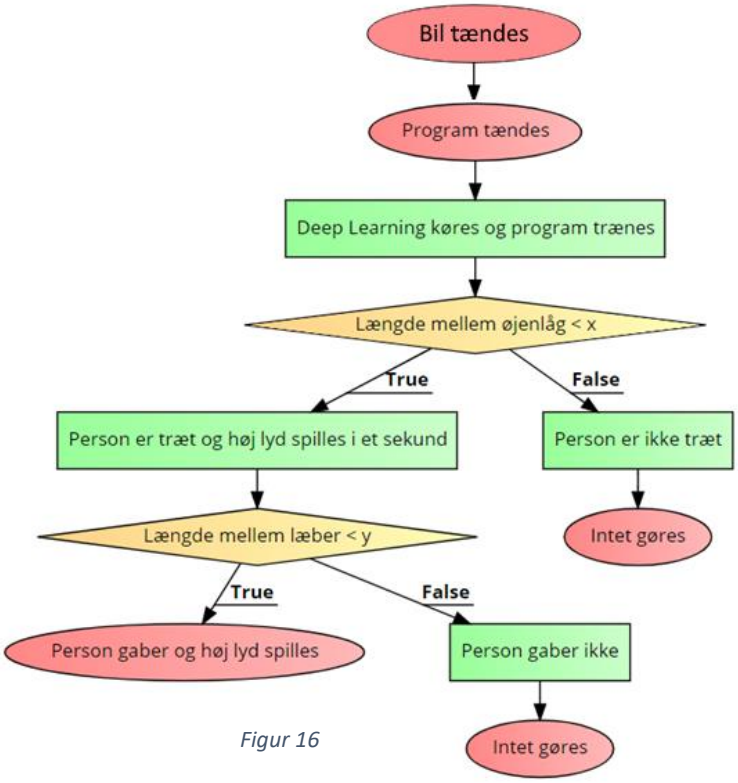
Når programmet skal finde ud af om en person er træt ved brug af munden tjekker den blot om afstanden mellem overlæbe og underlæbe er større end en grænseværdi. Hvis grænseværdien overskrides, betyder det at personen har munden åben og dermed gaber. Munden fungerer ligesom øjet som en kontakt. (Se figur 15)



Figur 15
Top-venstre: En visualisering af en mund der er lukket.
Top-højre: En visualisering af en mund der er åben.
Bunden: Visualisering af mouth-ratio.
Stigningen repræsenterer et gab.

7.3.4 Flowchart

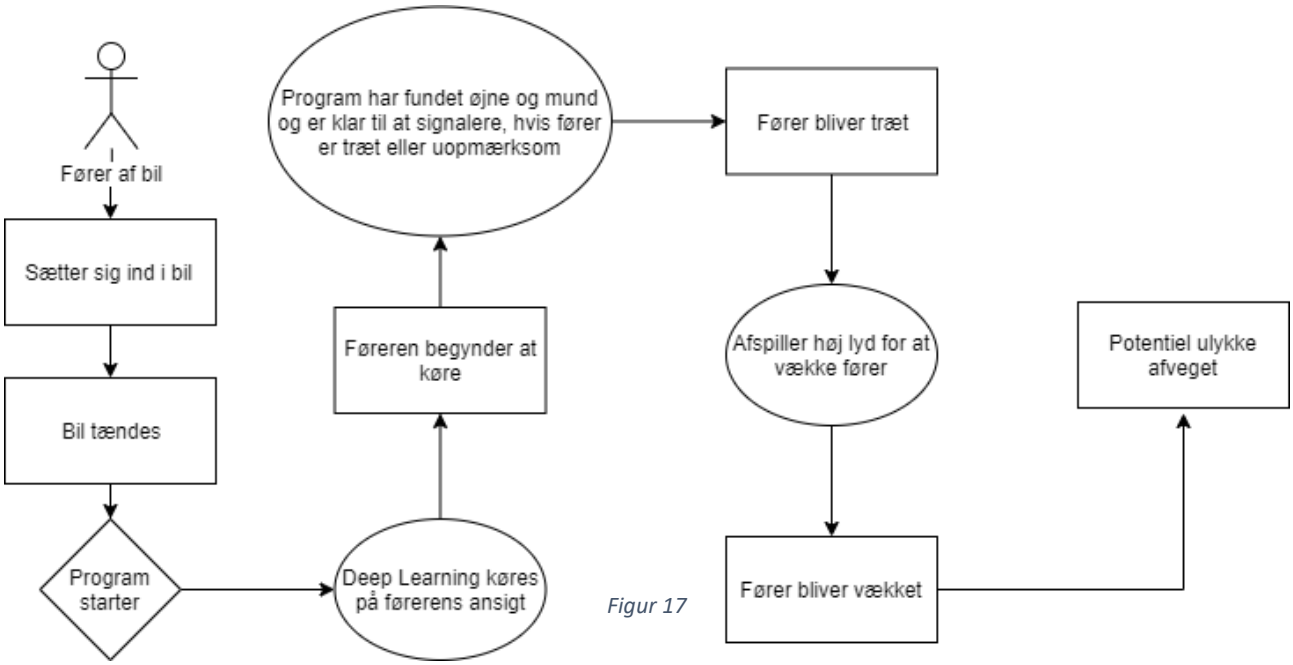
Et flowchart er en måde hvorpå man afbilleder processer og algoritmer, så man tydeligt kan se, i hvilken rækkefølge og orden ting forløber i. I figur 16 ses der hvordan programmet forløber.



Figur 16

7.3.5 Use-Case-Diagram

Figur 17 skal illustrere et brugerscenarie. Det tages udgangspunkt i at føreren af bilen, sidder normalt og ikke laver unormale ting og bevægelser under kørslen. Bruges produktet ikke på følgende måde, er der chance for at det ikke vil virke efter formålet.



Figur 17

Systemet skal altså hjælpe føreren af en bil blive opmærksom, hvis programmet ser behovet for det.

7.4 Implementering af produkt

Produktet er ikke til megen hjælp stående alene, produktet skal implementeres i en bil. I kommende afsnit diskuteres der hvordan produktet bedst kan implementeres.

Der er 3 måder hvorpå produktet kan blive installeret i bilen:

- Raspberry Pi
- Indbygget computer i bilen
- Apple carplay og android auto

Fælles for alle disse løsninger er at der skal installeres et kamera i en bil permanent, lidt lignende et bilkamera/dash camera. Samme løsning som (Rosebrock, 2017) også gør brug af i hans bil.

Figur 18 - (Rosebrock, 2017)

Rosebrock har en meget praktisk løsning til at implementere hans program. Han har blot en computer stående på passagersædet. Denne løsnings er praktisk, da en bærbar indeholder højtaler og kan køre programmet nemt. Dette vil ikke fungere i praksis, da der ikke vil kunne sidde en på passagersædet.

Derfor vil man kunne installere en Raspberry Pi tæt på USB Webcam'et og få Raspberry Pien til at køre programmet. Der er dog en masse restriktioner og krav til Raspberry Pien. Man kunne også installere programmet i Apple Carplay eller Android Auto.

Implementeringen af produktet skal kunne:

- Være i stand til at levere en masse processor kraft, eftersom programmet kræver meget.
- Programmet skal kunne tænde ved opstart og programmet skal ikke kunne slukkes af føreren
- Skal kunne gøre brug af kamera med usb-port.

7.4.1 Raspberry Pi

Her gennemgås fordele og ulemper ved de forskellige måder at implementere produktet.

Fordele

- Småt og håndterbart
- Kan udskiftes nemt, ved behov.

Ulemper

- Ingen nem måde at få lyd på
- Svag processor kraft
    - Det er nødt til at være den nyeste model og kraftigste model, da programmet kræver meget processor kraft. Måske ved brug af nyeste model kan programmet køre.
        - Nyeste model på markedet, Raspberry Pi 4 Model B
- Overopheder nemt, ved brug af program
- Kræver manuel installation

7.4.2 Apple Carplay – Android Auto

Det vil også være logisk at gøre brug af de nye integrerede Apple og Android instrumentbræt.

Fordele

- Har processer kraften til at køre programmet.
- Instrumentbrættet tændes ved opstart, og derfor bør det være muligt at køre programmet ved opstart af bil.
- Skal blot installeres på instrumentbrættet

Ulemper

- Skal skrive programmet om, da Android Auto og Apple Carplay ikke kan køre Python programmer
- Der kan kun laves Apple Carplay apps og programmer ved brug af Apple produkter og programmer.
- Programmet kan muligvis slukkes eller deaktiveres af brugeren
- Kan muligvis ikke understøtte et USB-kamera.

7.4.3 Computer installeret i bilen

Eftersom ovenstående implementeringer af produktet, ikke er optimale, kan der gøres noget lignende Adrian Rosebrock. Permanent installere en mini computer uden skærm, og uden alt det unødvendige. Denne løsning minder meget om en Raspberry Pi, blot med en normal computers specifikationer. På denne måde, ville man have processer kraften, man ville kunne køre Python programmet, og man vil kunne sætte programmet til at køre når der kommer strøm til computeren og den tændes.

Fordele

- Kan løse alle krav til produktet
- Kan komme med højtaler
- Har processer kraften til at køre programmet.

Ulemper

- Meget besværlig at installere
- Dyr
- Installationsprocessen vil være vanskelig

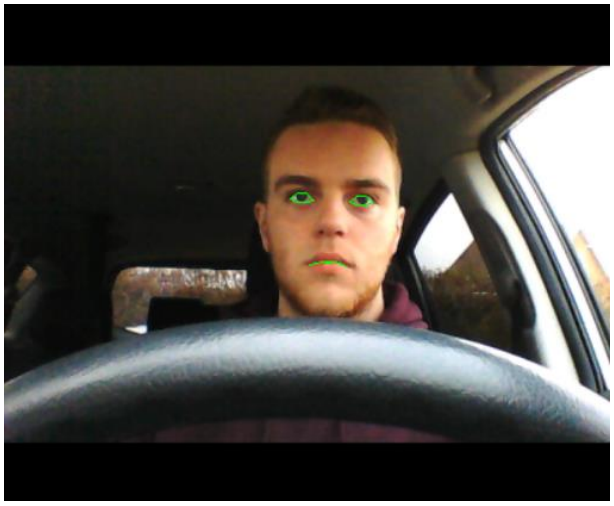
7.4.4 Del konklusion

På baggrund af ovenstående vurdering og analyse, vurderer jeg at, en computer installeret i bilen, vil fungere bedst. Dog, har jeg ikke kompetencerne eller ressourcerne til at installere en computer i en bil, hvilket er årsagen til at den anden bedste løsning, Raspberry Pien vil blive brugt.

7.5 Brugertest – Virker produktet faktisk?

Der ses nu den sidste fase af prototypen implementeret i en bil. Prototypen er i stand til at …

- Detektere hvor ansigtet befinder sig (figur 19),
- Detektere om øjnene er tæt på lukkede og dermed træt (figur 20).
- Detektere om personen gaber (figur 21)
- Detektere om personen kigger på sin telefon der ligger i skødet (figur 22)
- Detektere om personen gaber og har tæt på lukkede øjne (figur 23)
- Skelne forskel på om personen kigger i spejle (figur 24, 25, 26)



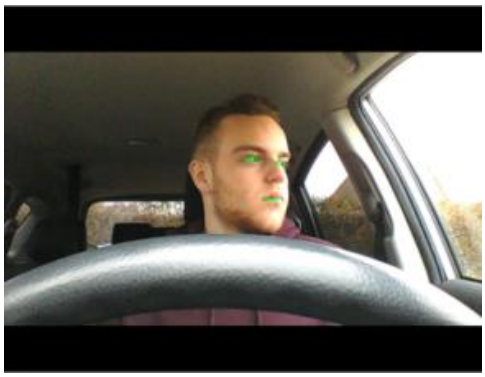
Figur 19 - Ansigtet kan ses

Figur 20 – Kan detektere om øjnene er tæt på lukkede

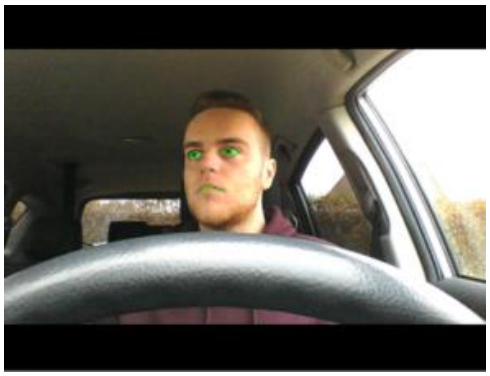
Figur 21 – Kan detektere om personen gaber

Figur 22 – Kan tjekke om folk er uopmærksomme i.e kigger på sin telefon der ligger i skødet

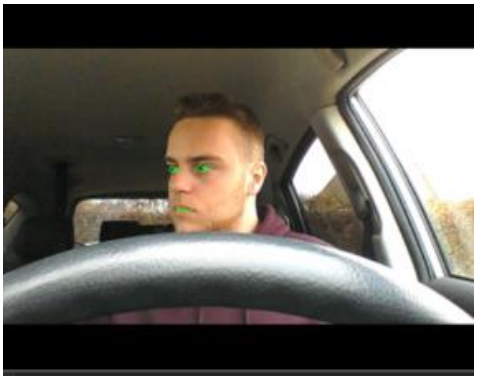
Figur 23 – Kan se øjne og gab på samme tid



Figur 245 – Siger ikke at personen er træt, hvis de kigger i venstre sidespejl



Figur 25 – Siger ikke at personen er træt, hvis de kigger i bakspejl



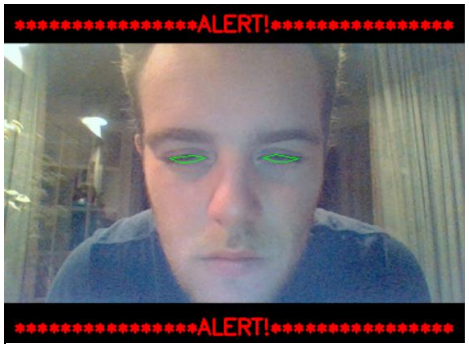
Figur 26 – Fortæller ikke at personen er træt, hvis de kigger i højre sidespejl

Figur 27 – USB-webcam installeret i bilen

7.6 Fordele/Ulemper og optimering af produkt

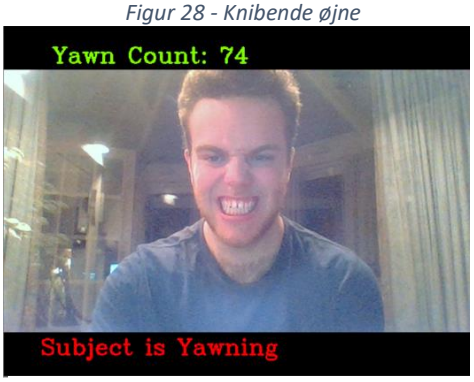
7.6.1 Ulemper

Ved dette produkt er der plads til forbedring og optimering ved mange punkter.
Der er fejl som, hvis personen har knibende øjne pga. solen skinner direkte ind på personen, vil programmet antage at personen er træt. Dette er grundet, at programmet udregner afstanden mellem øjnene.



Figur 28 - Knibende øjne

Har føreren af bilen en samtale kørende, vil føreren kunne blive opfanget som at gabe, hvis personen blot åbner munden tilstrækkeligt. I scenarier hvor føreren griner, vil programmet også sige at føreren gaber. Derudover, hvis man har en deformitet ved munden, vil programmet også have svært ved at finde munden, samt at vide hvornår personen gaber.



Figur 29 - Gab upræcist

Er føreren træt og tager hånden op foran munden, er det logisk at programmet ikke ville kunne tracke mundens bevægelse, og dermed ikke vide om personen gaber, dette gælder også for øjnene. Er der noget inde foran halvdelen af ansigtet, vil programmet ikke være i stand til at finde det ene øje. Dog, hvis det kun er det ene øje der dækkes, som med en klap for øjet, vil programmet stadig være i stand til at finde omridset af hvor øjet bør være, som ses på figur 30.

Vinklen hvorpå føreren af bilen sidder, er også en stor faktor, da dette betyder at kameraet skal tilpasses til føreren. Sidder føreren med hovedet skråt, vil programmet heller ikke være i stand til at tracke øjnene, samt munden.

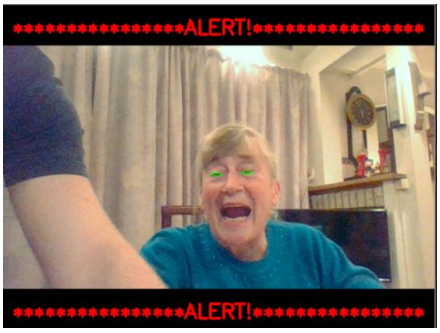

Figur 30 – klap foran øje, kan stadig forudsige hvor det skal være

Har føreren af bilen, solbriller på der indebærer enhver form for REVO-coating, vil det mindske præcisionen af produktet og programmet. Derudover, hvis en person har stærke asiatiske træk, vil produktet ikke være tilpasset til personen til at starte med, og vil vurdere at personen er træt i lang tid ved opstart. Derudover, vil maskering af store dele af ansigtet også medføre upræcis tracking.

7.6.2 Fordele

Produktet har tros mange fejl også mange funktioner der fungerer godt.

- Produktet er i stand til at blive brugt på enhver aldersgruppe.
- Briller har ikke nogen betydning for om personen kan bruge produktet, medmindre det er solbriller eller personens briller dækker for øjet.
- Er i stand til at finde ud af om en person er træt.
- Kan gøre føreren af en bil mere opmærksom, i tilfælde af træthed.



Figur 31 -Virker på ældre mennesker

7.6.3 Optimering

Hvordan kan programmet/produktet blive bedre.

- Programmet kan skrives i et andet sprog,
    - Kan med fordel skrives i C#, C++, C, sprog der er compilede sprog, fremfor Python som er et fortolket sprog. Compilede sprog kører generelt hurtigere, og bruger mindre processer kraft.
    - Dog er sværhedsgraden meget højere og det vil være svært at skrive programmet i disse sprog.
- Flere observationer
    - Der kan med fordel blive implementeret flere observationer. Programmet kigger på mund og øjne. Der kan blive tilføjet næse, øjenbryn, kinder osv. som kan øge præcisionen. Dog vil dette kræve en del mere processer kraft.
- Tilpasses så etniske forskelligheder ikke skaber problemer og upræcis tracking
    - Der kan laves en opstart proces for programmet, hvor den detekterer hvilken race personen er, og sikre sig at den tilpasser sig ud fra det. Dette betyder at den ikke vil forveksle en skandinavisk person med en kineser.
- Måden hvorpå programmet vækker føreren
    - Når programmet opfanger at føreren er træt, afspilles der en lyd der bliver højere og højere. Denne lyd kan muligvis ikke vække de trætteste bilister. Derfor kan en anden måde at vække på blive implementeret. Der kan blive skabt en forbindelse til sædet og programmet, hvilket vil betyde at, hvis personen ikke reagerer på høj lyd, kan sædet blive rystet, hvilket burde resultere i at føreren vågner. Dette kan dog også betyde at føreren mister kontrol over bilen.
- Flere faktorer
    - Programmet tjekker om munden er åben til en vis grad samt om øjnene er tæt på lukket. Der kan med fordel blive implementeret en øjne tracker, der vil danne en basislinje for hvordan føreren kører normalt. Denne basislinje kunne gemmes og derefter blive brugt til at finde ud af om føreren bliver uopmærksom eller noget andet er galt.

7.7 Teknologivurdering

Hvordan bliver produktet implementeret i samfundet?

Airbaggen blev implementeret i slut 1990'erne. Den blev implementeret ved at staten indførte et fradrag i registreringsafgiften på 5000 kr. pr. airbag. Dette betød at alle hurtigt fik anskaffet sig en airbag. (Ukendt, Den danske ordbog - Moderne dansk sprog, ukendt)

Med mit produkt kan det gøres på præcist samme måde.

Produktet og ideen vil blive pitched overfor politikere. Godkender politikerne produktet, skal det ideelt blive implementeret på samme måde som de gjorde med airbaggen, dog med træthedsgenkendelse. Siger de nej, vil medierne finde ud af det, og der vil komme dårlig omtale. Derfor skal politikerne have en god årsag til hvorfor de ikke vil implementere produktet. Kommer der ingen aftale kan man gå til andre metoder.

Andre metoder kunne være at komme i kontakt med forsikringsselskaber der er i kontakt med bilproducenter. Man kunne pitche ideen overfor forsikringsselskabet og få dem til at indgå en aftale med bilproducenter om at de skulle installere produktet i deres biler. Dette ville betyde at forsikringsselskabet skulle betale bilproducenten for at installere dem, dog ville forsikringsselskabet tabe færre penge, da der vil forekomme færre ulykker i trafikken som de skal dække med forsikringspenge.

8 Diskussion og Perspektivering

Hvorledes kan Deep Learning anvendes med fordele/ulemper i et overvågningssamfund?

Eftersom beskrevet i kapitel 6 er Deep Learning en måde hvorpå man bruger data til at forudsige resultatet af en given situation. Dermed kan enhver situation, hvor man kan indsamle data, og bruge denne data til at forudsige et resultat, være et punkt, Deep Learning kan automatisere.

Der er utallige punkter i et samfund, som kunne blive optimeret ved brug af Deep Learning. Dog er der også mange punkter som ville misbruge Deep learning til ondsindede eller for personlige gevinster.

8.1 Lufthavne

I lufthavne bliver der allerede nu brugt ansigts genkendelse til at se unormale hændelser hos mennesker og være præventiv til at se om der er en potentiel trussel. Derudover er det også begyndt at blive brugt andre steder i en lufthavn, som ved security check. I stedet for at bruge boarding pass, gør de brug af ansigts genkendelse til at finde ud af om det er den samme person, og derefter lukke dem ind i security check.

Deep Learning ansigts genkendelse i lufthavne	
Fordele:	**Ulemper:**
Øget sikkerhed	Kan hackes og krænkelse af privatliv
Forfalskning	Samtykke og uro
Hurtigere	Sammenspil med politi
Mindre behov for arbejdskraft	Potentiel skade
Nemt at implementere	Overvågning og sporing

Der er mange fordele, som vægter højt. Dog er der også mange ulemper, som for det meste skyldes menneskers mistro til systemer (Raedle, 2019) og hacker angreb på disse systemer, og deres frygt for at blive et mål for hackerne. I de fleste tilfælde vil dette ikke være et problem. Det er det eneste der er med til at holde udviklingen tilbage. (Ukendt, AURORA-AI, Ukendt)

8.2 Sundhedspleje

Deep Learning er i gang med at blive implementeret samt bliver det også allerede brugt i mange steder i sundhedsvæsnet. Det bliver brugt til at diagnosticere mennesker for sygdomme, og det vil også blive brugt til at hjælpe mennesker med at finde en kur til sygdommen. Derudover vil læger der har haft en meget lang vagt den givne dag, ikke have mulighed for at fejldiagnosticere.

Deep Learning har stort potentiale til at hjælpe mennesker, dog er der ulemper som gør folk bange for at bruge systemerne, såsom:

Deep Learning i sundhedspleje	
Fordele:	**Ulemper:**
Hurtigere	Kan hackes
Kan være mere præcis	Sammenspil med politi og overvågning
Nemt at implementere	Uro
Mindre behov for arbejdskraft	Potentiel skade
Mindre risiko for menneskelige fejl	

Databaser kan hackes, og i dette tilfælde betyder det at databaser der indeholder lister med mennesker der har visse sygdomme kan blive offentliggjort uden samtykke fra den syge. Derudover kan dette også blive brugt i sammenspil med politiet for at finde ud af om en person der har begået kriminalitet er kommet ind i systemet, og dermed kan de spore dem. (Ukendt, missinglink.ai, ukendt) (Kharkovyna, 2019)

8.3 Natural language processing – sprogteknologi

At have en samtale med en fra udlandet kan blive ufatteligt nemt, ved brug af en automatiseret oversætter. Personen fra det andet land, fører samtalen på deres sprog og den anden person vil føre samtalen på deres sprog. Dette vil gøre sprogbarrieren mindre og gøre kommunikation nemmere.

Deep Learning i sprogteknologi	
Fordele:	**Ulemper:**
Nedlægge sprogbarrierer	Kan gemme samtaler
Kommunikation bliver nemmere	Sammenspil med politi
	Besværlig at implementere og gøre præcis
	Udvikling i sprog, vil gøre den uprækcis
	Nødt til at gemme data

At være i et fremmedland, hvor man ikke snakker nationalsproget, vil det ved brug af NLP være nemt at få behandling, hvis man har behov i andre lande, da oversætteren blot vil gøre jobbet for en. (Elvis, 2018)

8.4 Del-konklusion

Deep Learning kan blive implementeret i mange aspekter af hverdagen, og vil optimere på alle faser der kan gøre brug af data. Teknologien er der, det er blot menneskenes syn på data og hacking der skal ændres før Deep Learning i hverdagen kan implementeres for alvor. Der ses de fem hyppigste og største fordele og fire hyppigste og største ulemper ved brug af Deep Learning:

Deep Learning overordnet set har disse fordele:	Deep Learning overordnet set har disse ulemper:
- Hurtigere - Mindre arbejdskraft - Mere præcis - Færre fejl - Øget sikkerhed	- Kan hackes - Kan misbruges af politi og staten - Kan blive brugt til at spore en befolkningsgruppe - Mennesker frygter at blive et mål for hacking, samt at deres personlige oplysninger bliver offentliggjort.

9 Sammenfattende konklusion

At mennesker sætter sig bag et rat, påvirket af træthed, medfører flere ulykker. Dette ses i mange statistikker fra respekterede organisationer, såsom, Havarikommissionen, Vejdirektoratet og Gjensidige Forsikring.

Deep Learning er en under-gren til Machine Learning og Artificiel Intelligence. Deep Learning er en måde hvorpå man bruger data til at forudsige resultatet af en given situation. Deep Learning formår at forudsige et resultat ved brug af metoder såsom, vægte, loss funktion, forward og back propagation og gradient descent. Disse metoder er med til at skabe et Kunstigt Neuralt Netværk, hvorpå man har en masse faktorer der spiller sammen, til at finde det endelige resultat. For at sikre sig at resultatet er korrekt gør man brug af back propagation, loss funktion og gradient descent for at tilpasse vægte der bliver ganget på en værdi. Det kunstige neurale netværk fungerer i store træk på samme måde som et menneskes.

Baseret på informationerne om ulykker forsaget af træthed, er der udviklet et produkt for at løse dette problem ved brug af Deep Learning. Produktet kan med sikkerhed opfange, hvornår en person er træt ud fra om personens øjenlåg er blevet tunge samt om de gaber. Produktet vil, hvis det har opdaget disse ting, vække personen eller gøre dem mere opmærksomme. Det gør den ved at udregne afstanden mellem nogle punkter programmet finder ved øjnene, og afstanden mellem overlæbe og underlæbe.

Produktet overholder opstillede krav, såsom ”Skal være sikkert og ikke være forstyrrende”, samt stemmer overens med GDPR. Produktet er ikke perfekt, hvilket betyder at der er plads til forbedring og optimering. Produktet fungerer på ældre mennesker, unge mennesker, samt hovedbeklædning i lille kalibrer og briller har ingen betydning for præcisionen af produktet.

Mit produkt er baseret på Deep Learning, dog kan det også bruges i mange andre scenarier i hverdagen, såsom sundhedssektoren, lufthavne og i sprogteknologi. Deep Learning kan blive implementeret i sundhedssektoren for at diagnosticere hvem der har hvilke sygdomme, fremfor at have en læge, der har haft en meget lang vagt gøre det. Brug af Deep Learning vil derfor øge succesraten og sikre at flere mennesker der kan behandles korrekt. Derudover kan man bruge Deep Learning til at opfange unormal adfærd der kan lede til terror i en lufthavn, og i applikationer for at oversætte mellem sprog. Før Deep Learning kan blive implementeret for alvor i verdens samfund, skal frygten for teknologi og hacking mindskes.

10 Referencer

Becker, D. (8.. Januar 2018). *Kaggle*. Hentet fra https://www.youtube.com/watch?v=wG6rdUURU-w&feature=emb_logo

Brandt, C. J. (23. Juli 23.07.2003). *netdoktor*. Hentet fra https://netdoktor.dk/sunderaad/fakta/traethed.htm

Brownlee, J. (1. November 2019). Hentet fra Machine Learning Mastery

Brownlee, J. (1. November 2019). *Machine Learning Mastery*. Hentet fra https://machinelearningmastery.com/what-is-deep-learning/

Cosic, U. &. (Juni 2008). Hentet fra https://www.ncbi.nlm.nih.gov/pubmed/18444356

David E. Rumelhart, G. E. (9.. Oktober 1986). *Nature*. Hentet fra Learning representations by back-propagating errors: https://ui.adsabs.harvard.edu/abs/1986Natur.323..533R/abstract

Elvis, U. (24. August 2018). *Medium*. Hentet fra https://medium.com/dair-ai/deep-learning-for-nlp-an-overview-of-recent-trends-d0d8f40a776d

EU. (25.. maj 2018). *GDPR*. Hentet fra https://gdpr.dk/

Fridman, L. (11. Januar 2019). *Youtube*. Hentet fra https://www.youtube.com/watch?v=O5xeyoRL95U

Gallup, K. (?. maj 2019). *Gjensidige Forsikring*. Hentet fra https://www.gjensidige.dk/

Hansen, C. (5.. August 2019). *mlfromscratch*. Hentet fra https://mlfromscratch.com/neural-networks-explained/#/

Hargave, M. (30. April 2019). *Investopedia*. Hentet fra https://www.investopedia.com/terms/d/deep-learning.asp

Hvid, I. &. (Maj 2019). *Trafikulykker for året 2018.* Vejdirektoratet.

Hvid, I. &. (Ukendt. Ukendt Ukendt). *Vejdirektoratet* . Hentet fra https://www.vejdirektoratet.dk/tema/udviklingen-i-trafikulykker

HVU. (2015). *Trafikulykker om natten.* Havarikommissionen for vejtrafikulykker.

Inc., M.-W. (19. Decebmer 2019). *Merriam-Webster* . Hentet fra https://www.merriam-webster.com/dictionary/machine%20learning

Jonassen, N. (15. December 2019). *Den Store Danske, Gyldendal*. Hentet fra http://denstoredanske.dk/index.php?sideId=34489

Justitsministeriet. (3. December 2013). *Justitsministeriet*. Hentet fra https://www.justitsministeriet.dk/nyt-og-presse/pressemeddelelser/2013/regeringen-skaerper-indsatsen-mod-spiritus-og-narkobilisme

Kessing, L. V. (14. Marts 2018). *Sundhed*. Hentet fra https://www.sundhed.dk/borger/patienthaandbogen/psyke/symptomer/traethed-en-oversigt/

Kharkovyna, O. (13. November 2019). *Towards Data Science*. Hentet fra https://towardsdatascience.com/artificial-intelligence-deep-learning-for-medical-diagnosis-9561f7a4e5f

Merriam-Webster, I. (19. December 2019). *Merriam-Webster Inc.* Hentet fra https://www.merriam-webster.com/dictionary/artificial%20intelligence

Nationnal Libray of Medicine National Institutes of Health. (1. September 2013). Hentet fra https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3738045/

Oppermann, A. (29.. September 2019). *DeeLearning Academy*. Hentet fra https://www.deeplearning-academy.com/p/ai-wiki-machine-learning-vs-deep-learning

Raedle, J. (8. Oktober 2019). *CNN*. Hentet fra https://edition.cnn.com/travel/article/airports-facial-recognition/index.html

Rosebrock, A. (8. Maj 2017). *pyimagesearch*. Hentet fra https://www.pyimagesearch.com/2017/05/08/drowsiness-detection-opencv/

Rumelhart, D. E. (1986). *Learning representations by back-propagating errors.*

Sharma, S. (9. September 2017). *towards data science*. Hentet fra https://towardsdatascience.com/what-the-hell-is-perceptron-626217814f53

Trafik, R. f. (September 2019). *Rådet for Sikker Trafik*. Hentet fra https://www.sikkertrafik.dk/raad-og-viden/i-bil/klip-i-koerekortet/klip-i-koerekortet-mobiltelefon

Ukendt. (Ukendt. Ukendt Ukendt). *AURORA-AI*. Hentet fra https://www.airport-suppliers.com/supplier/aurora-ai/

Ukendt. (ukendt. ukendt ukendt). *Den danske ordbog - Moderne dansk sprog*. Hentet fra https://ordnet.dk/ddo/ordbog?query=tr%C3%A6t

Ukendt. (ukendt. ukendt ukendt). *missinglink.ai*. Hentet fra https://missinglink.ai/guides/deep-learning-healthcare/deep-learning-healthcare/

12 Bilag

12.1.1 Kode med kommentarer

```
# importere moduler
from scipy.spatial import distance
from imutils import face_utils
import cv2
import imutils
import dlib
from sound import App
import numpy as np


# Definerer lyd fil, facedetection filer, threshhold.
sound = "beep.mp3"
thresh = 0.25
frame_check = 7
detect = dlib.get_frontal_face_detector()
predict = dlib.shape_predictor(".dat")


# Finder landmarks
def get_landmarks(im):
    rects = detect(im, 0)
    if len(rects) > 1:
        return "error"
    if len(rects) == 0:
        return "error"
    return np.matrix([[p.x, p.y] for p in predict(im, rects[0]).parts()])

# Plotter landmarks.
def annotate_landmarks(im, landmarks):
    im = im.copy()
    for idx, point in enumerate(landmarks):
        pos = (point[0, 0], point[0, 1])
        cv2.putText(im, str(idx), pos,
                    fontFace=cv2.FONT_HERSHEY_SCRIPT_SIMPLEX,
                    fontScale=0.4,
                    color=(0, 0, 255))
        cv2.circle(im, pos, 3, color=(0, 255, 255))
    return im


# Finder overlæbe
def top_lip(landmarks):
    top_lip_pts = []
    for i in range(50, 53):
        top_lip_pts.append(landmarks[i])
    for i in range(61, 64):
        top_lip_pts.append(landmarks[i])
    top_lip_mean = np.mean(top_lip_pts, axis=0)
    return int(top_lip_mean[:, 1])
```

```python
# Finder underlæbe
def bottom_lip(landmarks):
    bottom_lip_pts = []
    for i in range(65, 68):
        bottom_lip_pts.append(landmarks[i])
    for i in range(56, 59):
        bottom_lip_pts.append(landmarks[i])
    bottom_lip_mean = np.mean(bottom_lip_pts, axis=0)
    return int(bottom_lip_mean[:, 1])
# Finder ud af om munden er åben.
def mouth_open(image):
    landmarks = get_landmarks(image)
    if landmarks == "error":
        return image, 0

    image_with_landmarks = annotate_landmarks(image, landmarks)
    top_lip_center = top_lip(landmarks)
    bottom_lip_center = bottom_lip(landmarks)
    lip_distance = abs(top_lip_center - bottom_lip_center)
    return image_with_landmarks, lip_distance


# Funktionen der finder afstanden mellem øjnene.
def eye_aspect_ratio(eye):
    a = distance.euclidean(eye[1], eye[5])
    b = distance.euclidean(eye[2], eye[4])
    c = distance.euclidean(eye[0], eye[3])
    ear = (a + b) / (2.0 * c)
    return ear

# Definerer hvilke steder i ansigtet der bruges.
(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_68_IDXS["left_eye"]
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_68_IDXS["right_eye"]
(mouthStart, mouthEnd) = face_utils.FACIAL_LANDMARKS_68_IDXS["inner_mouth"]
cap = cv2.VideoCapture(0)
flag = 0

yawns = 0
yawn_status = False

while True:
    ret, frame = cap.read()
    image_landmarks, lip_distance = mouth_open(frame)

    prev_yawn_status = yawn_status

    # Hvis læbernes afstand er større end det her, så gaber personen.
    if lip_distance > 25:
        yawn_status = True

        cv2.putText(frame, " Subject is Yawning", (0, 400), cv2.FONT_HERSHEY_COMPLEX, 1, (0, 0, 255), 2)
        output_text = " Yawn Count: " + str(yawns + 1)
        cv2.putText(frame, output_text, (0, 100), cv2.FONT_HERSHEY_COMPLEX, 1, (0, 255, 127), 2)
```

```
    else:
        yawn_status = False

    if prev_yawn_status is True and yawn_status is False:
        yawns += 1

    frame = imutils.resize(frame, width=450, height=450)
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    subjects = detect(gray, 0)
    for subject in subjects:
        shape = predict(gray, subject)
        shape = face_utils.shape_to_np(shape)  # konverterer til NumPy Array
        leftEye = shape[lStart:lEnd]
        rightEye = shape[rStart:rEnd]
        mouth = shape[mouthStart:mouthEnd]

        # siger at leftEAR = funktionen kørt på variablen leftEye.
        leftEAR = eye_aspect_ratio(leftEye)
        rightEAR = eye_aspect_ratio(rightEye)

        # eye-aspect-ratio er derfor lig.
        ear = (leftEAR + rightEAR) / 2.0

        # Tager punkterne og forbinder dem.
        leftEyeHull = cv2.convexHull(leftEye)
        rightEyeHull = cv2.convexHull(rightEye)

        # Indikerer hvor munden, venstre og højre øje er.
        cv2.drawContours(frame, [leftEyeHull], -1, (0, 255, 0), 1)  # image, contours, contourIdx, color
        cv2.drawContours(frame, [rightEyeHull], -1, (0, 255, 0), 1)
        cv2.drawContours(frame, [mouth], -1, (0, 255, 0), 1)

        if ear < thresh:  # Hvis Eye-aspect-ratio er mindre end threshholdet, altså hvis øjet er knibet sammen.
            flag += 1
            if flag >= frame_check:  # Hvis flag er mindre eller ligmed frame_check så afspiller programmet lyd.
                cv2.putText(frame, "*****************ALERT!*****************",
                            (5, 30), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
                cv2.putText(frame, "*****************ALERT!*****************",
                            (5, 325), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
                App()  # spiller lyd
        else:
            flag = 0
    # Viser frames med live video feed.
    cv2.imshow('Live Landmarks', image_landmarks)
    cv2.imshow("Frame", frame)
    key = cv2.waitKey(1) & 0xFF
    if key == 13:  # Bliver der trykket enter, lukker programmet.
        break
cv2.destroyAllWindows()
```