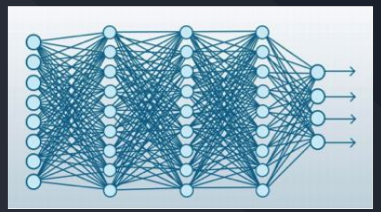
Selvkjørende bil ved hjelp av End-to-end learning



Ferdy Wessing
Fahd Boujmai
Julian Aaserud

Oppgavebeskrivelse

- Bruke Keras pakken samt andre pakker, det gitte rammeverket og bygge en CNN med egendefinert arkitektur til å lage en autonom bil.
- Bruke Udacity simulatoren til å generere treningsdata.
- Trene det nevrale nettverket på den innsamledet dataen
- Videoopptak av bilen som navigerer seg gjennom banen
- Rapportere resultatet

Mål med prosjektet

- Lage et nettverk basert på Dave-II arkitekturen som lar oss trene en bil på en vei og kjøre den på en annen vei, eller samme vei baklengs.
- Den ønskede oppførselen til bilen er at den skal:
  - holde seg så mye som mulig i midten av veien
  - Korrigere seg selv når den avviker fra midten
  - Kjøre stødig uten unødvendig svinging
- Eksperimentere med forskjellige arkitekturer og hyperparametre for å forbedre ytelsen

Relevant Litteratur

- om Dave-2 nettverket:
  - https://devblogs.nvidia.com/deep-learning-self-driving-cars/
  - https://arxiv.org/pdf/1604.07316.pd
- Bildeklassifisering - hvordan designe nettverket:
  - https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
- Implementere CNNs i Keras
  - https://github.com/keras-team/keras/blob/master/examples/mnist_cnn.py
- Bruk av Udacity simulatoren til dyp læring:
  - https://towardsdatascience.com/introduction-to-udacity-self-driving-car-simulator-4d78198d301d

Tidligere arbeid

Nvidia sitt **Dave-2** nettverk

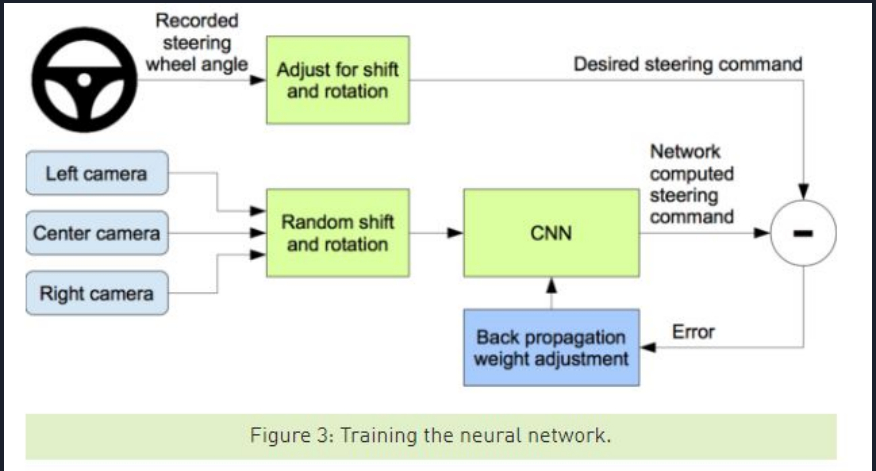


Figure 3: Training the neural network.

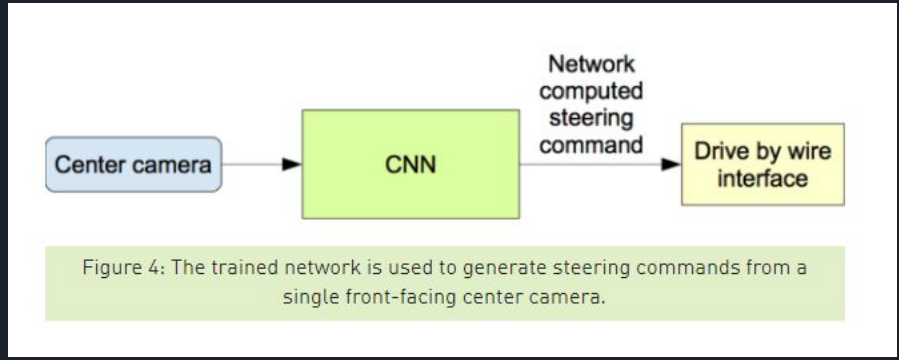
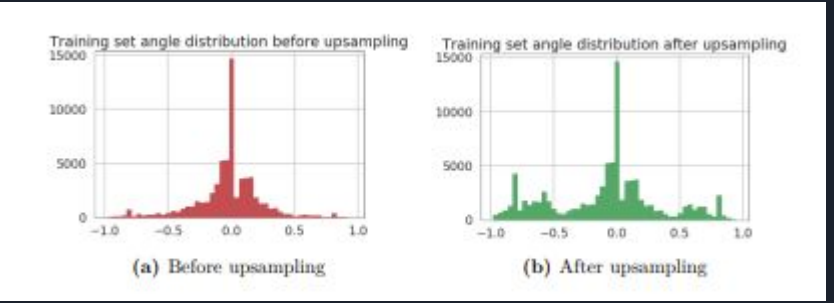


Figure 4: The trained network is used to generate steering commands from a single front-facing center camera.

https://devblogs.nvidia.com/deep-learning-self-driving-cars/

Tidligere arbeid



(a) Before upsampling (b) After upsampling

master skrevet ved NTNU av Anna Kastet og Ragnhild Cecilie Neset

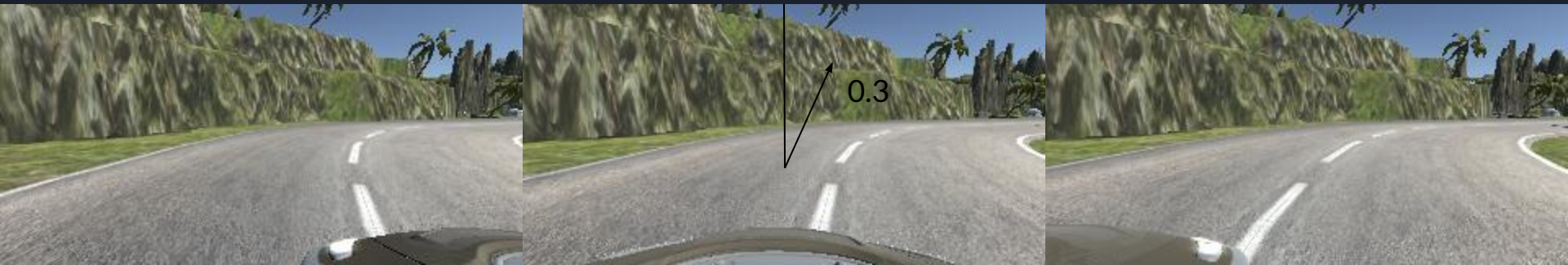
- Forbedret evne til å detektere “dårlig kjøring”

Layer type	Kernels@Kernel size	Stride size	Output size	Activation
Input	-	-	3×200×60	-
Conv2D	24@5×5	2×2	31×98×24	ReLU
Conv2D	36@5×5	2×2	14×47×36	ReLU
Conv2D	48@5×5	2×2	5×22×48	ReLU
Conv2D	64@3×3	1×1	3×20×64	ReLU
Conv2D	64@3×3	1×1	2×18×64	ReLU
Flatten	-	-	1152	-
Fully Connected	-	-	100	ReLU
Fully Connected	-	-	50	ReLU
Fully Connected	-	-	10	ReLU
Fully Connected	-	-	1	Linear

https://brage.bibsys.no/xmlui/handle/11250/2566504

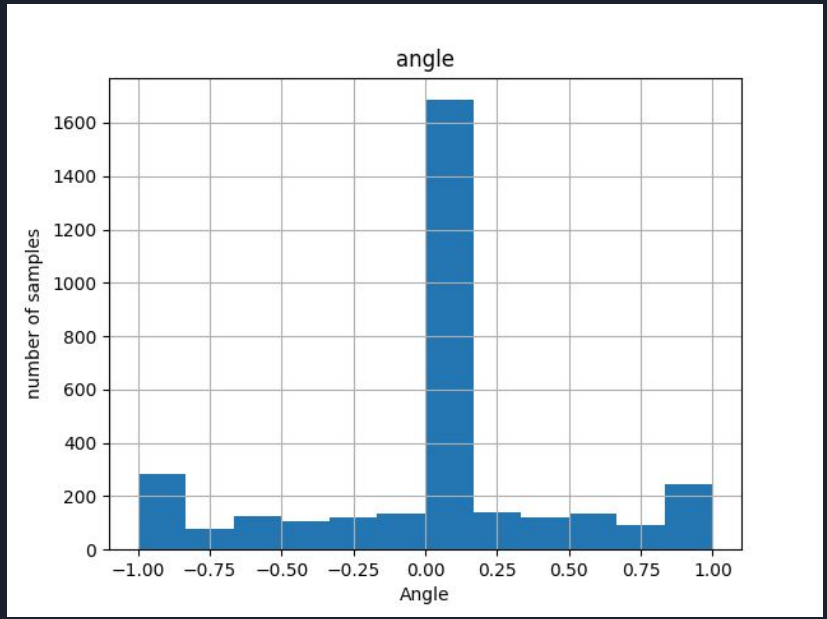
Datasett

- Vi bruker simulatoren til å generere et datasett med bilder fra de tre kameraene foran på bilen. Disse bildene parres opp med tilsvarende registrert styrevinkel.
- Dataen blir så brukt til trening og validering av modellen.
- Alle tre bilder er fra samme øyeblikk, og med én tilsvarende styrevinkel.
  I dette tilfellet 0.3



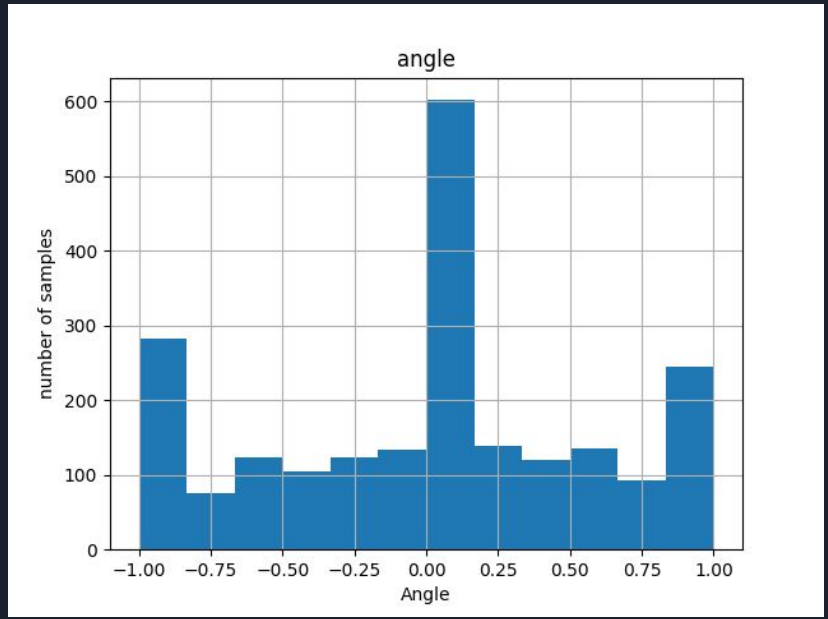
Pre-prosessering av data

- midt-kameraet inneholder for mye data med en styrevinkel på 0
  - å bruke dette resulterer i et bias for å kjøre rett frem
- Et nettverk som velger å alltid kjøre rett frem vil ha veldig lite loss på datasettet



Pre-prosessering av data

- Løsning: vi fjerner 70% av dataen med styrevinkel lik null.
- Ønsker fremdeles en stor mengde “rett frem” data for å unngå unødvendig svinging.
- Potensielt problem: Kjøring med et tastatur kan medføre et dårlig datasett.
    - Har som vane å gi enten null pådrag på styring eller max pådrag på styring.
    - Mulig at et ratt eller spillkontroller hadde gitt mer jevn endring i pådrag til styrevinkel.

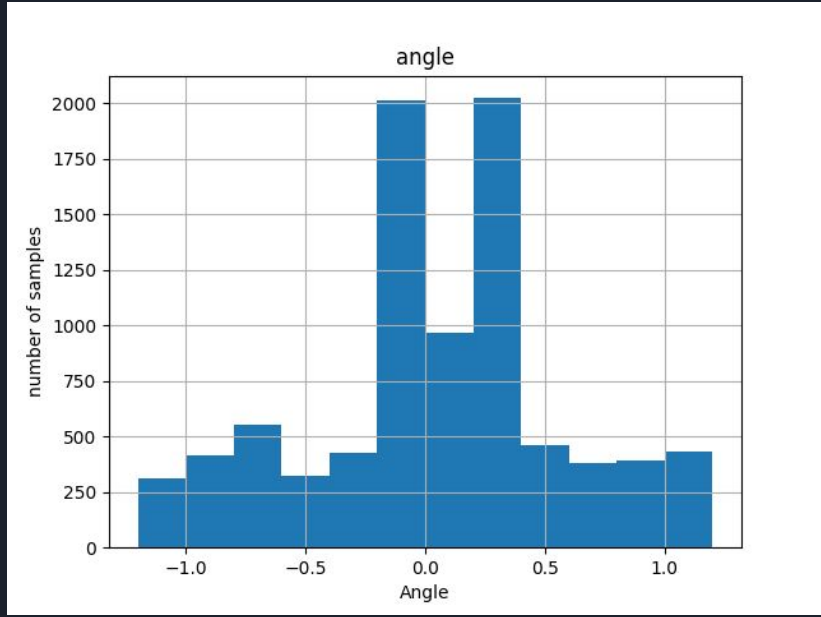


Pre-prosessering av data

- Bruker i tillegg ett venstre og høyre kamera for å lære bilen å svinge tilbake til midten
- Gjøres ved å legge til en kompenserende vinkel som styrer den mot midten
- Ønsker å beholde store deler av dataen, da den skal klare å kjøre tilbake mot midten overalt
    - kan gi problemer med for mange verdier som har akkurat de kompenserende vinklene

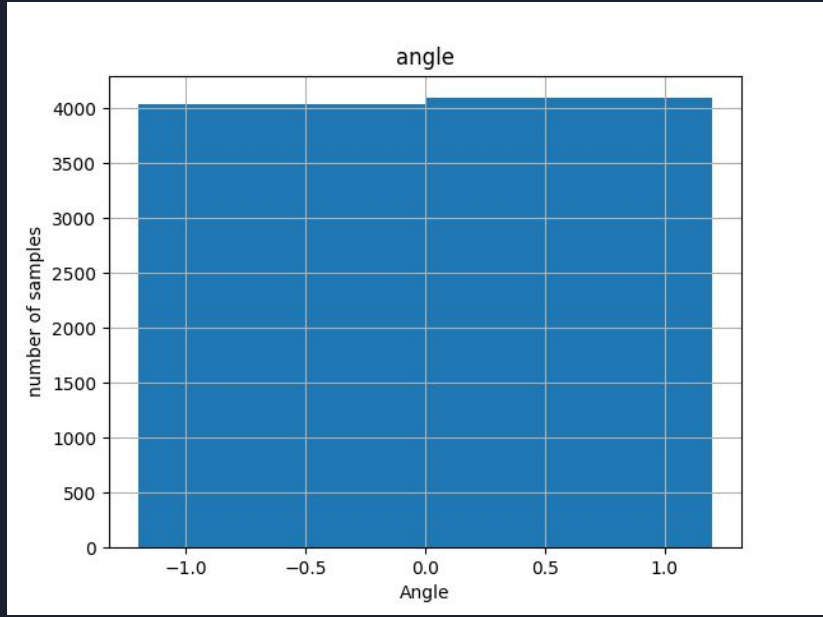
Pre-prosessering av data

- Sette sammen venstre, senter og høyre kamera til et datasett
- Datasettet skal fokusere på å holde bilen på midten av veien
- Noe bias mot kjøring rett fram for å unngå slingring



Pre-prosessering av data

- Fjerner all dataen der bilen kjører rett frem (styrevinkel = 0)
- Viktig at datasettet er balansert, slik at den ikke gjør det betydelig dårligere i svinger en vei
- Ser at datasettet er ganske balansert



Data Augmentasjon

- Den eneste data augmentasjonen som brukes er endring av **lysstyrke**
  - Generaliserer datasettet til forskjellige veier og lys-settinger
  - Gjort ved å konvertere bildene til HSV (Hue, saturation, value) for så å endre value parameteren, som bestemmer lysstyrken
  - Multipliserer den originale verdien med et tilfeldig tall mellom 0.5 og 1.25
  - begrenser verdiene til å være mellom 0 og 255 for å unngå ugyldige verdier for lysstyrke
  - Endres tilbake til BGR-formatet etterpå

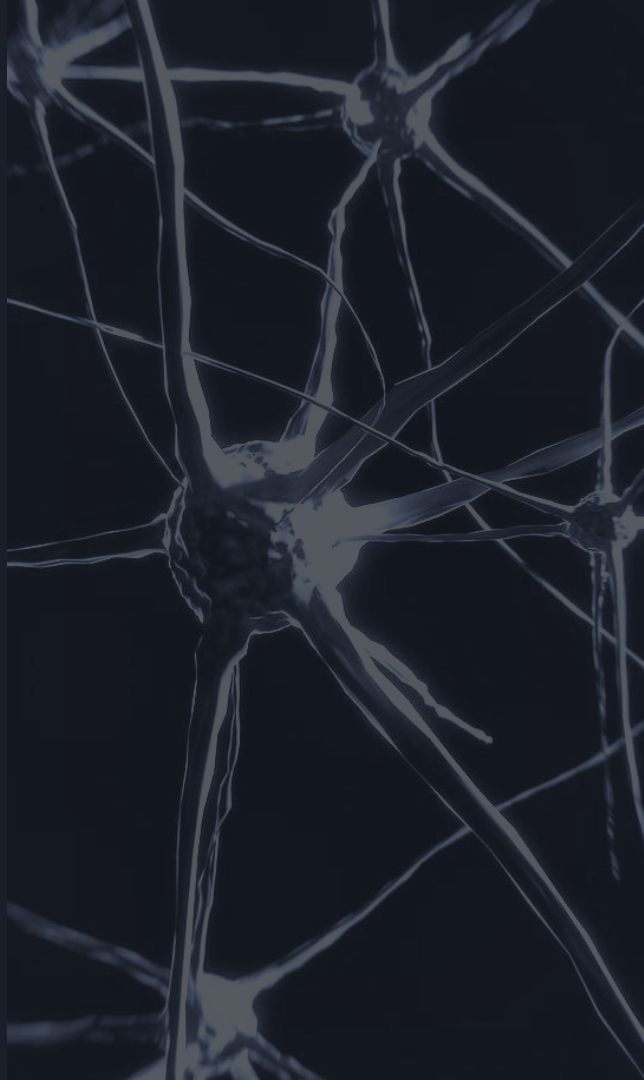
Data Augmentasjon resultater

Original:

tilfeldige endringer:

Hyperparametre

- Bildedimensjoner: = (66, 200, 3)
- Offset til sidekamera: = 0.3
- Læringsrate: = 0.001
- Decay til læringsrate: = 0.01
- Optimaliseringsfunksjon: = Adam
- Tapsfunksjon: = Mean squared error
- Epoker: = 15
- Batch størrelse: = 100
- Trening/validering splitt: = 5%
-



Network Architecture

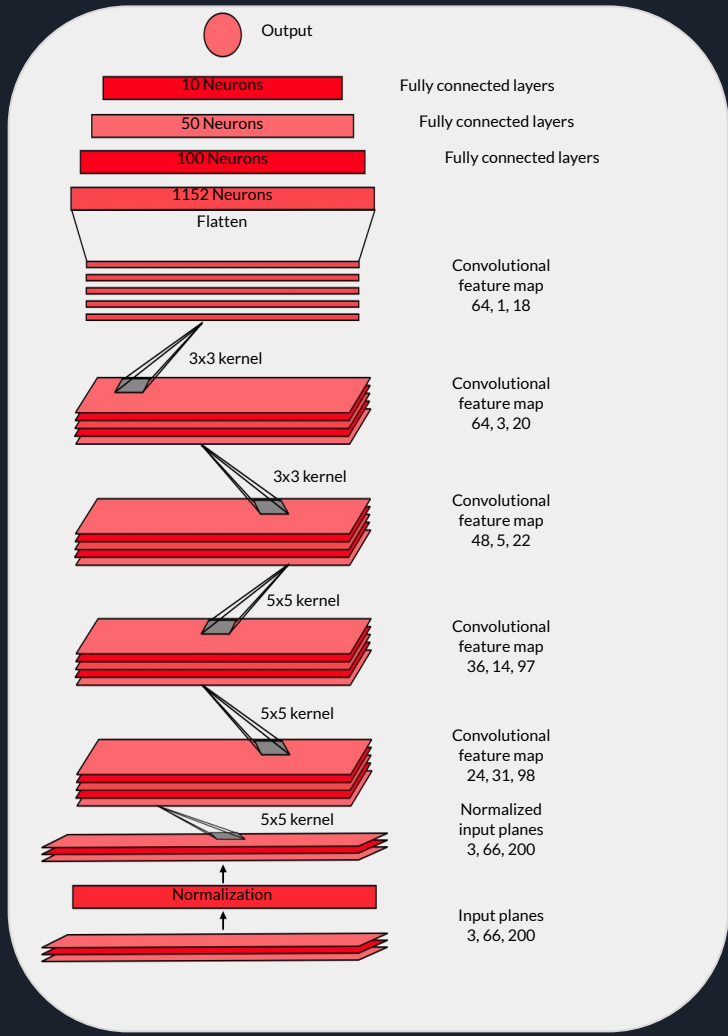
- Bildet til høyre er DAVE-2 arkitekturen vi har basert nettverket vårt på
  - Antall parametre: 253,815
  - Trenbare parametre: 253,017
  - ikke trenbare parametre: 798

- Vi bruker en modifisert variant av denne. Vi har:

  - Vi har gruppenormalisering etter hvert lag

  - Dropout

  - Bruker ReLu aktiveringsfunsjon etter hvert konvolusjonslag



Midtveisresultatene

- Bilen kjører autonomt i 15 mph
- kjører over halvparten av begge banene
- Klarer ikke alltid broa på bane 1, da den ikke har sett lignende data før
- Kræsjer på bane 2 når den ser en annen vei rett frem mens den er i svingen

Arbeid som gjenstår etter midtveisinnlevering

- Teste andre nettverksarkitekturer og data augmentasjoner for å forbedre opptreden
  - Legge til flere konvolusjonslag
  - Eksperimentere med forskjellige størrelser på filter, steglengde og feature map
  - Legge til dropout
  - Eksperimentere med flere og større fully connected lag
  - Eksperimentere med forskjellige optimaliseringsfunksjoner
- Generere en video av opptreden
- Eksperimentere med antall epoker

- **Hovedfokus: lage et dypere nettverk som tillater bilen å kjøre raskere og “tryggere”.**

Hovedproblemet med oppgaven

- Loss funksjonen er et dårlig mål på hvor bra kjøringen er
- Har ikke klar nok definisjon av hva vi vil ha av kjøringen
- Må veie flere ting mot hverandre:
  - Hvor god er den til å være midt på veien?
  - Hvor god er den til å holde seg på veien?
  - Må den kjøre uten å svinge unødvendig?
  - Vil man heller ha en bil som kjører veldig bra mesteparten av tiden, men ikke klarer en sving? eller vil man ha en bil som kjører “ok” overalt?

Endret pre-prosessering

- hvor mye “rett frem” data skal beholdes?
  - beholde for mye: bilen begynner å svinge for sent
- skal vi fjerne høyre og venstre kamera også når bilen kjører rett frem?
  - Fører til unødvendig svinging
  - kan skyldes at datasettet inneholder mest ingen styrevinkel og maksimum styrevinkel
- Beholdt pre-prosesseringen vist tidligere
  - Ga mest stabil kjøring, spesielt ved høyere hastigheter

Resultatet

- Bilen på track 2:
  - https://drive.google.com/open?id=17b3JVfsv6h0M5WozTHyzMg4AidyKy9F4
  - Kjører hele banen
- Bilen på track 1:
  - https://drive.google.com/open?id=1BMlbuWKYV_I4YgLHTouB11nb-oIrwQCP
  - Kræsjer når den kommer til “ukjente” omgivelser
- Mesteparten av tuning kom av å tilpasse bilen til 20 MPH
  - ingen “fartsdata” i treningssettet
  - må være mer aggressiv

Testing av ulike arkitekturer og data augmentasjon

- Parametere som vi har forsøkt å endre:
  - styrevinkelen lagt til på venstre og høyre kamera
  - flere fully-connected lag
  - Endre størrelsen på feature maps i konvolusjonslag
  - Ulike optimizers

Offset = 0.4

- Må reagere raskere grunnet høyere fart
- kompenserende vinkel økt fra 0.2 til 0.4
- funket best når all data fra høyre og venstre kamera ble beholdt
    - vil at den skal gå mot midten uansett hvor den er på banen

Ulike arkitekturer

- La til et ekstra fully-connected lag etter konvolusjonslagene
  - størrelse på 750
  - Setter seg ikke fast der den ser andre veier - men kjører likt ellers
    - kan være tilfeldig
  - er 10 ganger så mange parametre
- Endre feature map størrelse på konvolusjonslagene
  - Endret filterstørrelsen i de to siste lagene til 128 og 256
  - opplevdes som marginalt bedre

Optimaliseringsfunksjoner

- SGD med momentum:
  - klarte ikke å holde seg i midten av banen
- nAdam:
  - Vanskelig å tune tilstrekkelig, 3 parametere som må tunes
- Adam:
  - fungerer generelt best, ikke like sensitiv for tuning

Forbedring av datasettet

- Brukte 2 runder rundt banen
  - mer robust mot dårlig kjøring i datasettet
- Forsøkte å bruke et datasett bestående av begge baner
  - Nettverket klarte ikke lære seg å kjøre
  - loss-funksjonen virket bra

Epoker

- Originalt: 15 epoker
- Forsøkte å øke til 25 epoker, i tilfelle den kunne forbedres
  - Mindre generalisert, kjørte dårlig på bane 1
  - overfitting: siden datasettet bestod av mest data med styrevinkel på 0.4 kjørte den svingete heller enn å kjøre rett frem
- Ved mindre enn 15 kjørte den ikke like bra
- **Resultat: beholdt 15 epoker**

Konklusjon

- End-to-end læring
  - Vanskelig å “forstå” nettverkets avgjørelser for et menneske
  - vanskelig å gjøre robust - vet aldri når noe i omgivelsene får bilen til å reagere
- Tastatur gir dårlig datasett
  - Tendens til ingen eller maksimum styrevinkel
  - bør heller bruke et ratt eller kontroller med analog input
- Dave-2
  - Godt utgangspunkt for selvkjørende biler
  - Vanskelig å gjøre store forbedringer uten å radikalt endre arkitekturen
- Et stort og bra datasett sannsynligvis nøkkelen for å oppnå bedre resultater
  - Kan “kjenne igjen” veiskilt, autovern når det ses mange ganger