UNIVERSITY OF TARTU

Faculty of Science and Technology

Institute of Computer Science

Data Science Curriculum

Kristjan Roosild

Driving Speed as a Hidden Factor Behind Distribution Shift

Master's Thesis (15 ECTS-credits)

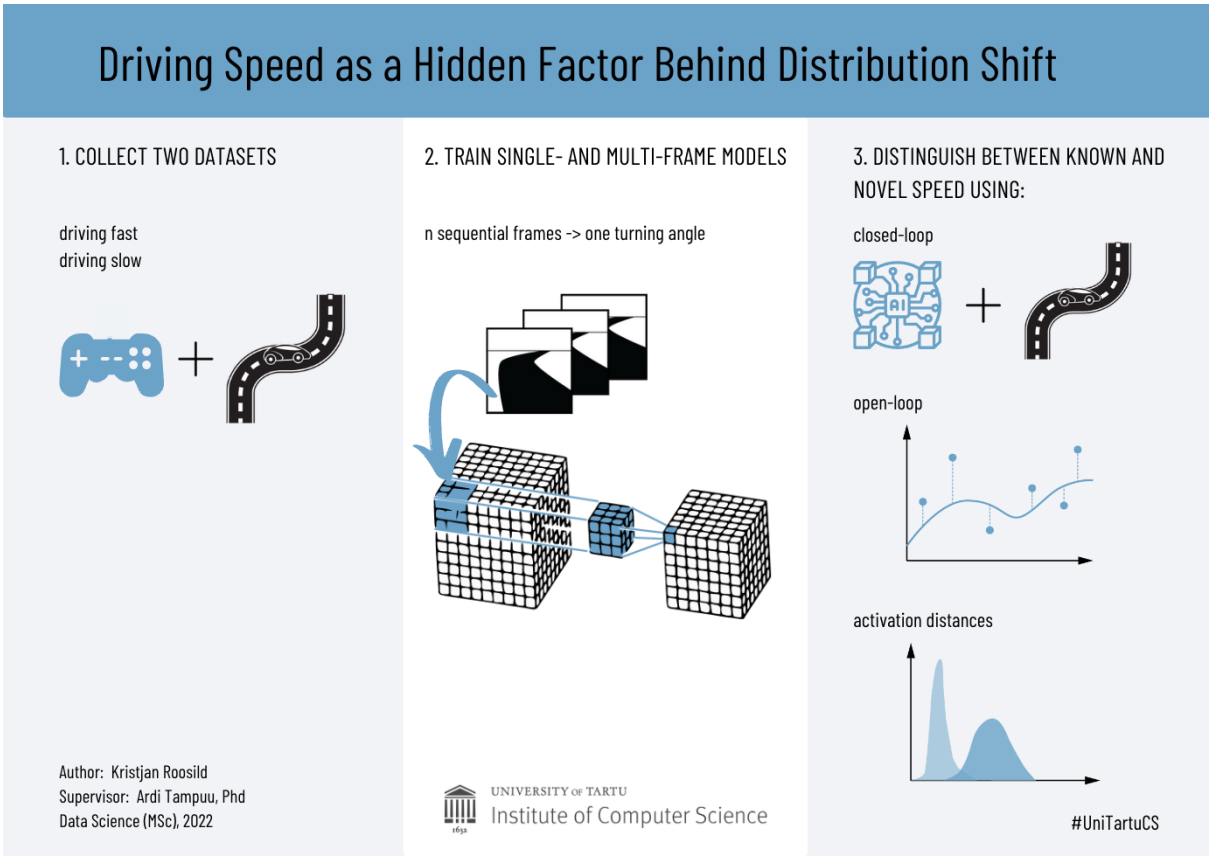
Supervisor: Ardi Tampuu, PhD

Tartu 2022

Driving Speed as a Hidden Factor Behind Distribution Shift

This work investigates the effects of using a self-driving neural network on data generated at a different driving speed, e.g. when testing a model at low speed for safety reasons. Using Donkey Car, an autonomous toy car, we find that weak performance at deployment may indeed be caused by driving at an unsuitable speed. Driving at a novel speed makes the system behave worse via two mechanisms. Firstly, a change in appropriate outputs (e.g. turning angle) happens, as faster driving requires more aggressive turning. Secondly, for models using multiple consecutive frames as input, different speed results in out-of-distribution inputs, which are known to be troublesome for networks.

These findings show the importance of testing and deploying self-driving systems by using the speed known to them.



Keywords

Autonomous vehicles, end-to-end self-driving, neural networks, model evaluation, out-of-distribution, distribution shift

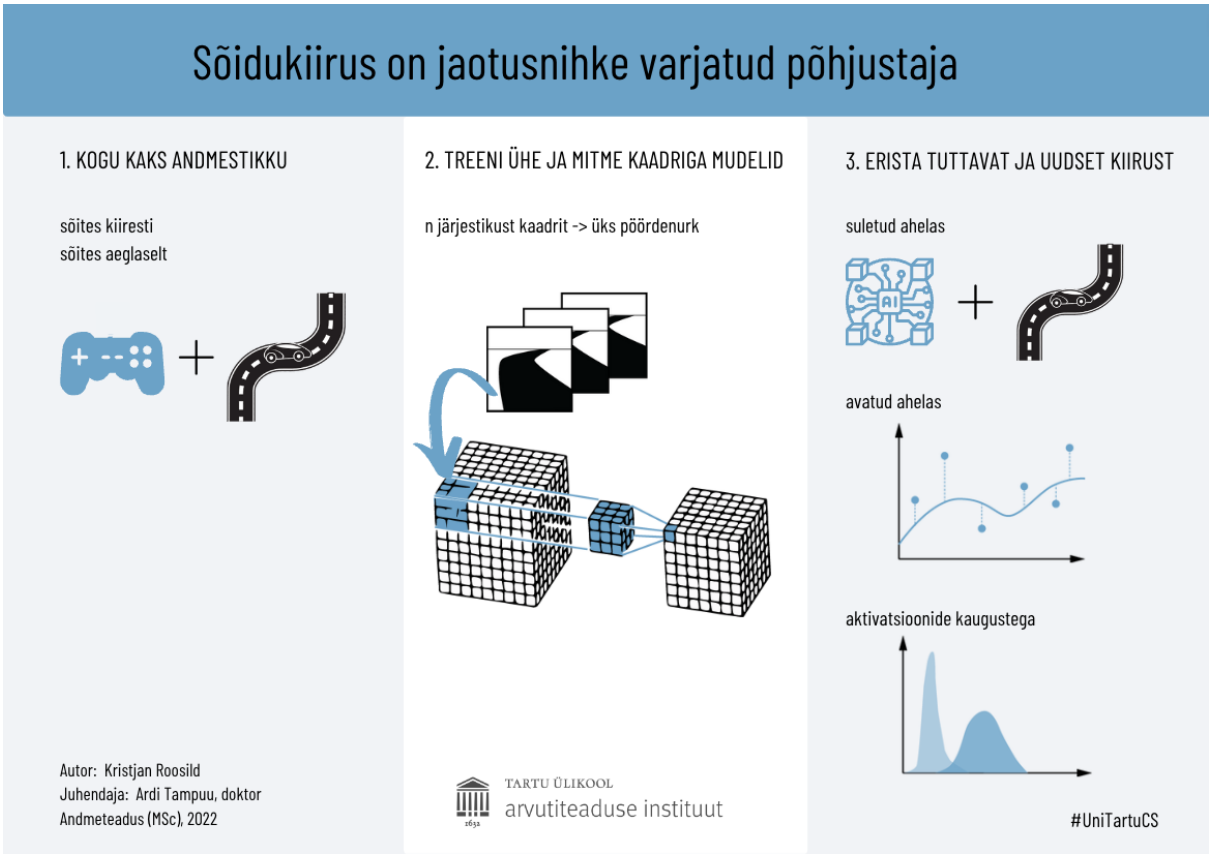
CERCS

P176 Artificial Intelligence

Sõidukiirus on jaotusnihke varjatud põhjustaja

Antud töö uurib erineval sõidukiirusel genereeritud andmete mõju isesõitvatele tehisnärvivõrkudele, näiteks juhul kui turvalisuse eesmärgil testitakse mudelit madalatel kiirustel. Autonoomse mänguauto Donkey Car abil leiame, et halva sõitmisoskuse põhjuseks võib tõepoolest olla ebasobiv kiiruse valik. Sõitmine kiirusel, mis on süsteemi jaoks uudne mõjutab seda negatiivselt läbi kahe mehhanismi. Esiteks, kiirem sõit toob kaasa agressiivsema keeramise mis tähendab väljundite muutust. Teiseks, mitmekaadriliste mudelite puhul on uudse kiiruse näol tegemist jaotusvälise sisendiga, mis on teada probleem tehisnärvivõrkudele.

Uuringu tulemused näitavad kui oluline on isesõitvate süsteemide testimine ja rakendamine neile teadaolevatel kiirustel.



Võtmesõnad

Autonoomsed sõidukid, otsast lõpuni isejuhtimine, tehisnärvivõrgud, mudeli hindamine, jaotusvälisus, jaotusnihe

CERCS

P176 Tehisintellekt

1 Introduction

Advancing the level of autonomy for self-driving is beneficial for humankind. It could mitigate congestion, reduce casualties and help decrease energy consumption. It could also increase productivity by enabling humans to spend their time doing other tasks than driving. The benefits are to be up to 800 billion dollars by 2050. [Montgomery]. Full autonomy can only be reached while keeping safety as a priority. Any accident caused by an autonomous driving system undermines trust and could cost lives. There is still much to do before we can reap the benefits. Although fully automated driving under limited conditions (SAE level 4) should start this year [Waymo], there is still a long way to go before reaching level 5. The weather conditions, environment and surrounding human behaviour are difficult to predict [Yurtsever].

Research in end-to-end self-driving has become a growing trend in autonomous vehicle research as opposed to modular approaches [Tampuu 2020]. In end-to-end driving the whole modular pipeline is replaced by a neural network.

The most used input data for end-to-end driving is monocular camera image. Using camera vision is the cheapest and road signs are made to reflect or emit visible light. It was already used by [Pomerleau] in 1989.

However, a self-driving end-to-end system that is safe for its user and others on the road needs to derive the speed of the commuters moving in its vicinity. It has to consider temporal aspects. Therefore, when using camera vision for sensing, it might be beneficial to use multiple camera frames to make one decision.

However, [Wen], [Wang], [LeCun], and [Bansal] showed that models predicting from observation histories perform worse than those that predict from a single-frame alone. Also the current Autonomous Driving Lab end-to-end network solution on a real car (Lexus) uses a single image (single video frame) for inferencing the steering command [Tampuu 2022]. It has been proposed by [Haan] that access to more information leads to worse generalisation performance. Causal misidentification occurs when inputs include historical information - for example multiple frames are used as input to predict a steering angle.

Beyond causal confusion, in this thesis we demonstrate that bad results at deployment may be caused by driving at an unsuitable speed. Notice that deploying models at a low driving speed is a common practice for safety reasons (e.g. [Tampuu 2022]). When driving at different speeds, the inputs and expected outputs of an end-to-end model may differ:

- If consecutive frames are considered as input, the difference between these frames is more significant when driving faster.
- The differences are amplified at higher speeds if we consider "change image"/optical flow as input.

- A change in speed causes a change in the desired steering angle (i.e. output) distribution due to an inevitable lag (e.g. actuator delays) in commands taking effect (at higher speeds, the model needs to predict turn commands earlier).

As a result of above observations, we hypothesise that multi-frame model performance at inference time suffers significantly if the deployment speed is different - out of distribution (OOD) compared to training data. The input data and the labels collected at speed X might not allow the model to drive at 0.5X or 2X. At those speeds, the model would:

1. encounter input data it has never seen before (e.g. consecutive frame difference increases with speed, resulting in image sequences the model has never seen before);

2. predict commands too early or too late as it attempts to counter lag at speed 1X, not at the current speed.

The second point holds for any models, including single-frame models.

1.1 Contributions

The contributions of this thesis are the following:

1. We collected data at fixed driving speeds and trained the neural network to imitate steering on the collected data based on camera image(s) (steer=f(images)).

2. We then show that prediction (open-loop) performance is better for an in distribution (InD, same speed) test set than an out of distribution (OOD, different speed) set.

3. We show that actual driving performance is better when driving speed is InD.

4. Finally, we show that multi-frame-input networks are destabilised by wrong speed inputs and hence the drop in performance is likely not solely caused by a change in appropriate outputs.
   Using the network's second-to-last layer's neuron activations, we demonstrate that OOD inputs cause clearly distinguishable activations from InD inputs, suggesting that the distribution shift of the inputs is also a factor in worse open-loop and closed-loop performances. The activations resulting from training speed and novel speed data stay clearly separate in tSNE embeddings and can be almost perfectly separated by a mahalanobis distance based classifier.
   We repeat the same experiments with the fast driving speed data synthesised from the slow driving speed data and witness similar results.

1.2 Outline

Background gives a short overview of the advantages and disadvantages of modular and end-to-end driving approaches. Two main learning methods for end-to-end approach are briefly described - reinforcement- and imitation learning. A succinct overview of the Donkey Car, the platform used for this work, is given.

Methodology provides a brief overview of how the hardware and software was used and configured. Data gathering and model training is described. Finally, a description is provided on how the analysis was conducted - in other words - how the results were produced.

Results first validates our assumptions about data and then use four different approaches to show that multi-frame model performance at inference time suffers significantly if the speed is out of distribution.

Conclusion reiterates the thesis goals, contributions and results, listing the possible future research.

2 Background

The following briefly introduces modular and end-to-end autonomous driving approaches and two main methods of training an end-to-end system. Next a Donkey Car self-driving platform is described. Finally the open- and closed-loop evaluation is described.

There are two main approaches to achieving autonomous driving - modular and end-to-end. During recent years the end-to-end approach has become a viable contender to modular methods in autonomous driving [Tampuu 2020], [Yurtsever].

2.1 Modular Approach

The modular approach is widely used and considered conventional [Tampuu 2020]. Modularity comes from interconnected self-contained modules (Figure 1, Modular pipeline) such as perception, localization, planning and control [Yurtsever].

The most significant advantage of these systems is that developing individual modules divides the task of automated driving into different, more straightforward problems [Ch]. Another significant advantage is interpretability - it is easy to track an unexpected behaviour to the initial source of error [NTS Board] because interfaces between modules are human-designed and hence human-understandable.

One of the most significant disadvantages of modular systems is being prone to error propagation [McAllister]. Individual modules' predefined inputs and outputs might not be optimal in all required scenarios [Zeng]. Different road and traffic conditions require attending to different pieces of information from the environment. Using the example in [Tampuu 2020], the perception module usually provides 3D bounding boxes of relevant objects as an output. Information not contained in this representation is not retrievable for modules dependent on the perception module's output. A ball rolling onto a street in a residential area should cause the vehicle to slow down. Sudden braking for the same ball on a highway would be highly risky, and the odds of a pedestrian appearing to fetch it are slim. The perception module must include the ball among the detectable objects. Moreover, depending on the context, the planning module must react differently to a ball on the road. It should assign the ball a different cost depending on whether in a residential area or highway. There are a lot of similar, rare but essential scenarios, and a lot of engineering effort is needed to cover all of them [Dosovitskiy].

2.2 End-to-end Approach

[Tampuu 2020] wrote that more recently, end-to-end driving emerged as an alternative to modular approaches. This approach treats the entire system of transforming sensor inputs to driving commands as a single learning task (Figure 1, End-to-end pipeline). A deep neural network model trained with an end-to-end approach should learn optimal intermediate representations to solve the target task. There are no human-defined intermediate inputs/outputs, and the model can attend to implicit sources of information.

While the lack of intermediate outputs in the end-to-end driving approach is a promising solution towards achieving fully autonomous driving [Bojarski], then at the same time, it makes it much harder to trace the initial causes of errors [Zengg].

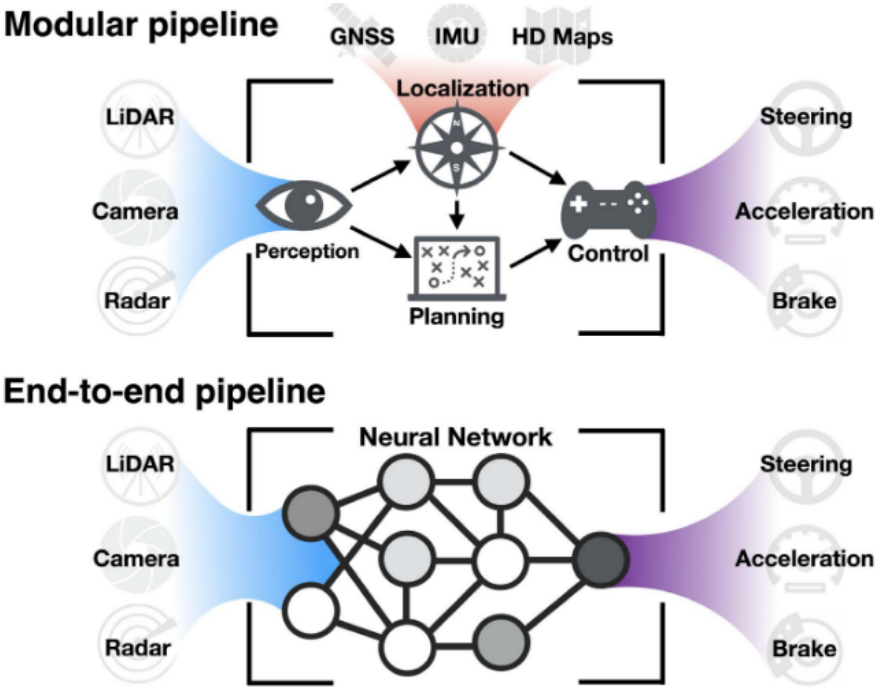


Figure 1. The differences of modular and end-to-end approaches. The modular approach is a pipeline of connected modules, while the end-to-end approach replaces the pipeline with one neural network [Tampuu, 2020].

2.3 End-to-end Training Methods

There are two methods to training an end-to-end autonomous driving system: reinforcement learning (RL) and imitation learning (IL).

2.3.1 Reinforcement Learning

With RL, the system learns to maximise the rewards or minimise the punishments of driving correctly or incorrectly. During online learning, the system encounters multiple driving situations and thus learns to deal with diverse scenarios. There is no need to collect labelled data but compared to IL, reinforcement learning is less data efficient, which means more time for training is needed. Doing RL in the real world is dangerous and requires a safety driver being present at all times, which, given the time it takes, could be more expensive. [Tampuu 2020]. [Kendall] demonstrated that it is possible to train a system to follow a single lane paved road with no traffic in a day using RL in the real world.

2.3.2 Imitation Learning

IL is a form of supervised learning where expert behaviour is mimicked by the model. It is used more prominently than RL in end-to-end driving. Usually, the expert behaviour is a human driver's commands like steering, acceleration and braking. The abundance of training

data makes IL work well on simple tasks like lane following. Rarely occurring and complex traffic scenarios are still a challenge for this approach as there is by definition less data about them to learn from. IL also suffers from the distribution-shift problem [Codevilla], [Ross]. When deployed, self-driving may take the vehicle into situations which the expert driving never encountered. The model underperforms in those situations as these are outside the distribution of training data [Tampuu 2020].

2.4 Donkey Car

The platform used in this thesis is called Donkey Car. It is an open-source self-driving platform with an end-to-end approach where the neural network is trained via imitation learning method. It consists of a remote-controlled miniature electric-powered toy car with Raspberry Pi, a single wide-angle fish-eye Camera and Python framework, supporting Keras and Tensorflow machine learning framework [DonkeyCar].

A pre-built four-wheel-drive Donkey Car model S1 (Figure 2) is sold by [RoboCar Store] with a Logitech F710 Wireless Gamepad. The gamepad has an analogue joystick to steer the car while gathering training data.

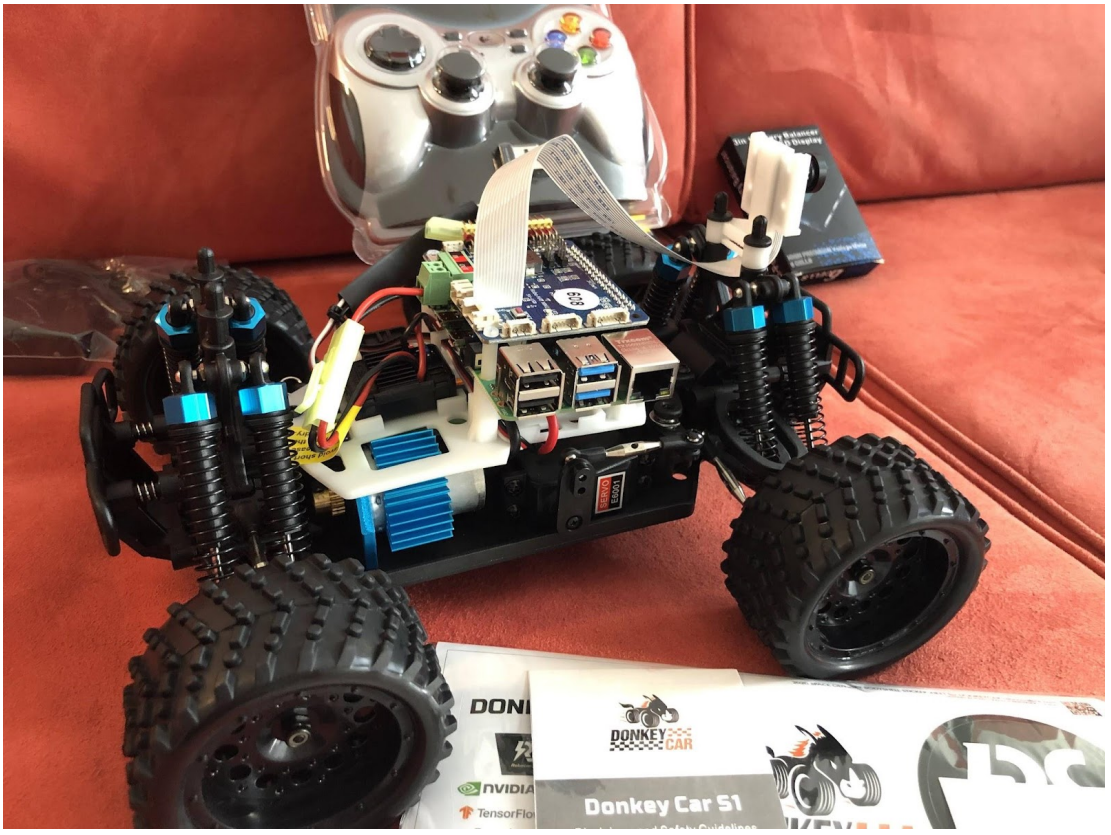


Figure 2. The car and the remote.

The platform is successfully used for education at hands-on events and [Maison du libre] and at competitions both in real life [DeltaX], [Renault], [DIY Robocars 2022-1] and in virtual environment [DIY Robocars 2022-2]. It has allowed for some interesting results already in University of Tartu by [Kurniawan] and [Uduste].

2.5 Open- and Closed-loop Evaluation

An open loop (also known as a feedforward control loop) is a control loop that has an absence of feedback (or the feedback is not used), while a closed control loop (also known as a feedback control loop) uses feedback - the next observation (or the state of the environment) depends on its current output.

In the case of an autonomous vehicle using an end-to-end pipeline, an open-loop evaluation means that the feedback loop of the neural network model's prediction to the world it operates in is open - its input at time t+1 does not depend on its predictions at time t as these predictions are not actually used. A part of the original dataset is used to evaluate the model and no actual driving is done. Similarity between predicted and ground truth values is measured.

Closed-loop in the case of an autonomous vehicle evaluation means that the feedback loop of the model's predictions to the environment is closed. The model is deployed onto a vehicle and the model's predictions are used as driving actions. The action taken at time t affects the input at time t+1. The driving behaviour is measured.

Some of the possible inputs are video feed from cameras, LiDAR data and current speed. Some of the outputs which are measured could be throttle value and steering wheel angle or actual speed and angle of front wheels.

The open-loop evaluation is easier, safer and cheaper to manage but the open-loop predictions only loosely correlate ($r > 0.6$) with closed-loop metrics [Codevilla]. Hence, it is necessary to perform closed-loop evaluation in addition to open-loop evaluation.

To lower safety risks, it is common to evaluate closed-loop performance on real cars at lower speeds compared to the speed the human was driving while collecting training data. [Kendall] was driving at maximum of 10 km/h. [Tampuu 2022] did closed-loop evaluation with 80% of the data collection speed. In another instance, The Autonomous Driving Lab in Tartu University tested the performance of their Lexus car on the road by driving at 30 km/h whereas the model training data was collected at the speed the driver was comfortable with [personal communication]. [Aidla] uses 50% of the human driving speed.

The fact that often the trained model's performance is evaluated at a novel speed makes it even more important to know if and how much this affects the performance of a model.

3 Methodology

3.1 Platform

Donkey Car model S1 (Figure 2) was used for this work. A default 20 Hz setting was used where in each second the system captures and saves 20 images (frames), steering angles and throttle value. The 8-bit RGB colour images have 160x120px resolution. During training and inferencing, the top part of the image is cropped as shown on Figure 3. This way, the network learned to predict only based on what it sees on the track and not on the background, achieving better generalisation and avoiding distraction caused by objects like people standing close to the track.

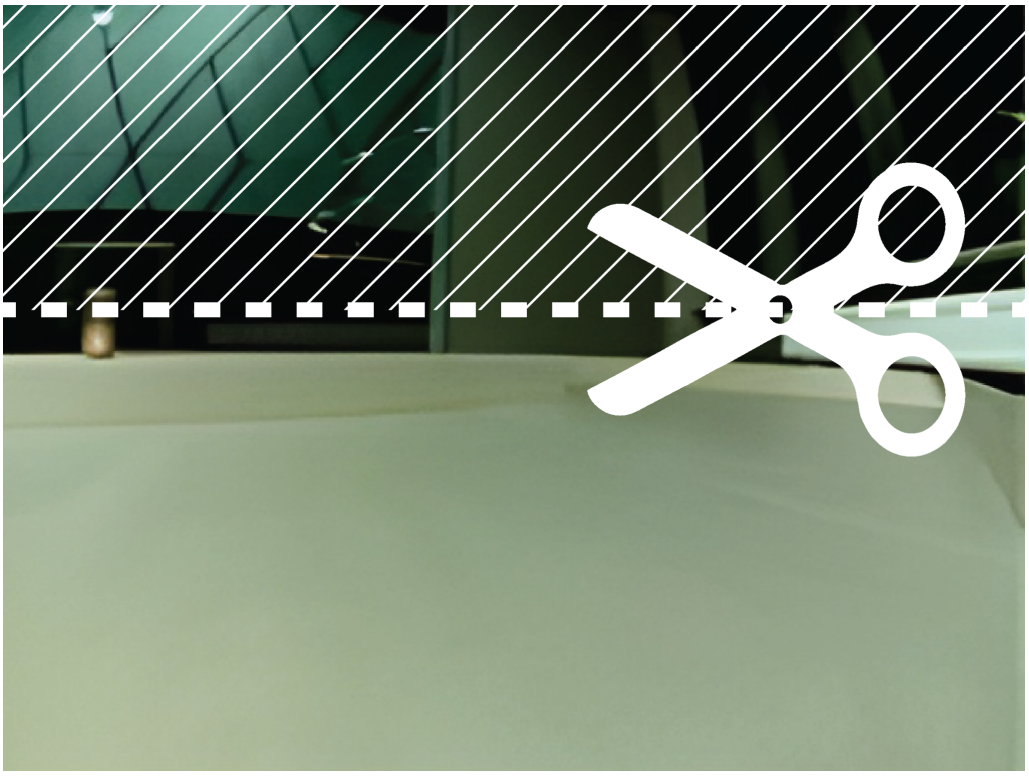


Figure 3. Cropping the top part of the image.

3.2 Track

The training data was gathered and closed-loop testing was performed on the track shown on Figure 4. Occasionally, when driving on the track, the car would hit the walls and move them out of place. To avoid any accidental changes with the track between training and testing a double-sided tape was used and the planks were glued to the floor. Data was gathered driving counter-clockwise only. The track resides immediately next to a big window. The lighting conditions at daytime were very different compared to nighttime. There were some especially strong reflections on the track during daytime.

Figure 4. The track used for data gathering and closed-loop testing.

3.3 Neural Network Architecture

Two network architectures were used. A network that predicts the turning angle based on only one frame (single-frame model) and a network that uses multiple consecutive frames (multi-frame model) to predict the turning angle.

The chosen single-frame network architecture (Figure 5) is the preferred one in DonkeyCar documentation and also showed excellent performance at [DeltaX] in the 2022 competition, used by the winning team [Abdumalikov]. A more detailed description of this architecture is described in Appendix II, Table 7.

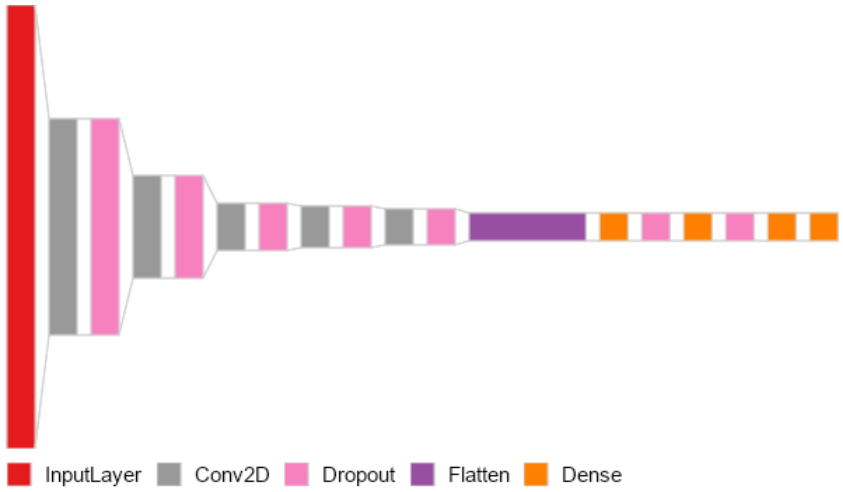


Figure 5. A general view of layers of the single-frame network architecture used in this work.

There are two types of model architectures which can handle multiple past inputs, CNN+RNN and Fixed window CNN [Tampuu 2020]. For this work, fixed window CNN (Figure 6) was chosen because the RNN architecture inference time tested too big and by that affected the closed-loop performance. The multi-frame model uses three images for each angle prediction. Hence the speed effect is still noticeable while the inference time is low enough not to affect the driving performance. A more detailed description of this architecture is described in Appendix 8.2, Table 8.

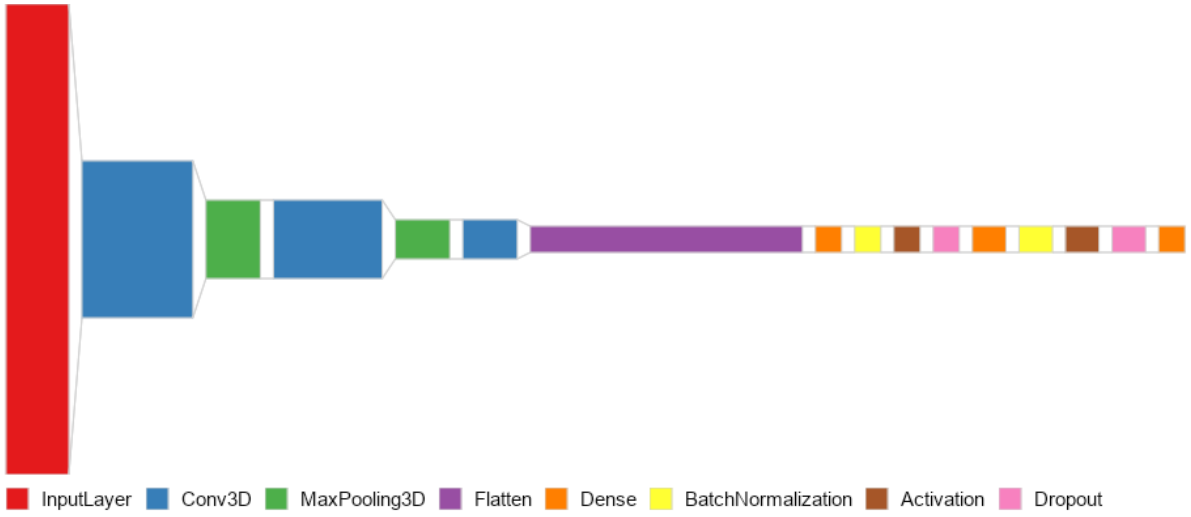


Figure 6. A general view of layers of the multi-frame network architecture used in this work.

3.4 Data Gathering

3.4.1 Automated Data Gathering

During the first round of data gathering a human controlled the steering. However, the angle predicted by the resulting model proved to be smoother than the ground truth angle of a human (Figure 7). Initial closed-loop testing proved that AI could still drive without hitting

the track wall with these smoother turns. To reduce unwanted effects from rough human driving, a robust single-frame first-generation (the teacher) model was trained using multiple driving speeds so it would be capable of driving in a wide speed range. The teacher model was then used to drive autonomously on the track and gather the training data with smoother turns. That training data was then used for second-generation (student) model training. The results are based on the latter.

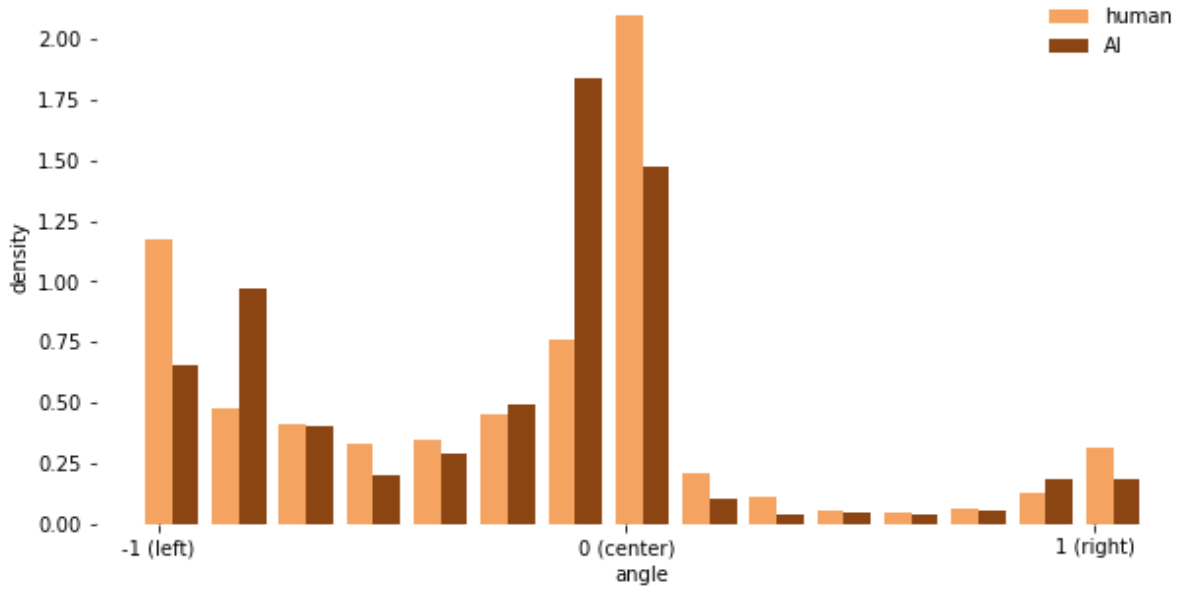


Figure 7. Human driving results in more extreme ground truth turning angles.

3.4.2 Constant Speed

The Donkey Car framework provides the ability to set and increase-decrease a constant throttle value using the remote controller. However, a constant throttle value does not guarantee a constant car speed. There are two reasons for it. Firstly, the car slows down during turning - the steering servo motor causes a decrease in voltage while in actuation. This was countered by improving the driving algorithm: when the absolute turning angle value grows, then increase throttle. Secondly, speed starts to decrease over time due to electronic speed controller (ESC) overheating. Thirdly, battery depletion results in reduced voltage and speed. These effects were countered by counting the seconds per lap and increasing the constant speed setting as needed while the car was driving.

It is interesting to note that ESC heated up faster while driving slowly and not while driving fast, which would make more intuitive sense. It may be caused by the smoother turns while driving slowly. During slow driving, the servo motor might be used more causing the ESC to heat up faster.

The initial data collection was done for three different throttle values. It became apparent that it is difficult to distinguish between medium and low throttle values as the resulting actual speed was very similar. This led to the decision to have the teacher (first-generation) model re-gather all the data with only two different driving speeds. As an added bonus of only two

speeds, fewer types of data resulted in fewer student (second-generation) models to train and analyse which in turn made the results easier to present and grasp.

From here on, to be more succinct while referring to the datasets, the dataset which has been recorded while driving with the fast speed (14,85 s/lap) will be referred to as "fast data". The dataset which has been recorded while driving with the slow speed (24,25 s/lap) will be referred to as "slow data".

3.4.3 Driving Direction

A counter-clockwise driving direction was chosen because the car's left turning radius is shorter than the right turning radius. As with driving speeds, this decision resulted in fewer different datasets to manage and fewer models to train.

3.4.4 Time of Day

The goal of this work was not to build a robust model which would be able to generalise well for lighting conditions and drive both during the day and night. To minimise changing lighting conditions and strong reflections on the track all driving was done at night-time.

3.4.5 Data Cleaning

Datasets were cleaned by removing the records where the car touched or crashed into the side of the track. A simple web application called Tub Clean (Figure 8), built specifically for cleaning these datasets, was used. It allows playing back the recorded frames as a video and marking regions of frames as deleted.

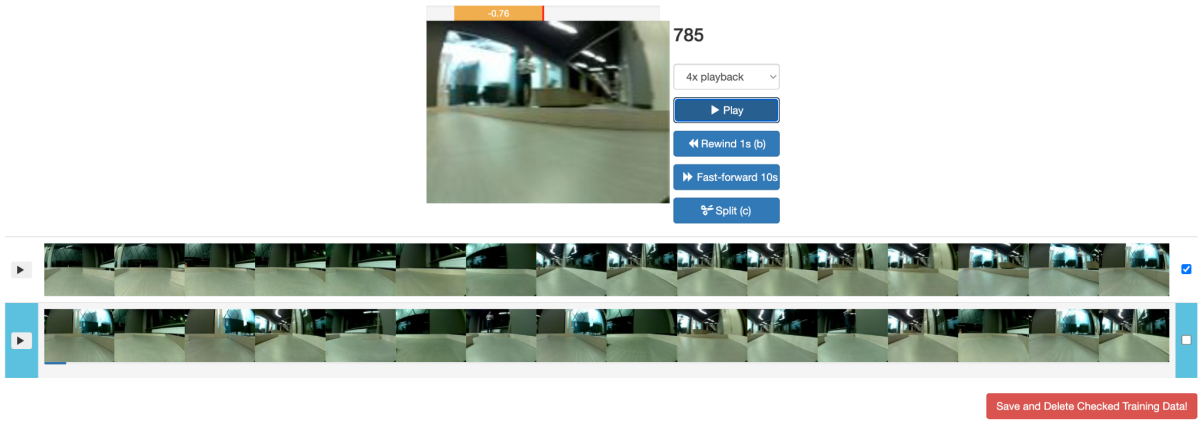


Figure 8. Tub Clean web application, part of the Donkey Car open source framework.

After cleaning, the fast data size is 19,250 frames, and the slow data size is 20,304.

3.5 Training

Four models types were trained:

1. Single-frame architecture, trained on fast data.

2. Single-frame architecture, trained on slow data.
3. Multi-frame architecture, trained on fast data.
4. Multi-frame architecture, trained on slow data.

Fast and slow data were split into five folds. Four folds were used as training and one fold as test data. Models with every possible combination of four folds were trained, resulting in 4x5=20 models in total. The model weights were saved as h5-files and not as Tensorflow Lite (tflite) files. Tensorflow Lite files are smaller and result in almost twice as fast inference time, but lack the support of 3D convolutional layers.

Training multi-frame models where each sample has three consecutive frames and constructing folds with samples assigned randomly would result in an unacceptably big portion (96%) of target leakage.

Example of random sample assignment with 21 samples and three folds:

Training samples in the first fold: 1 3 4 5 6 7 8 10 13 16 17 18 19 20

Test samples in the first fold: 2 9 11 12 14 15 21

In the example, the samples 1 and 3 are in the training set and sample 2 is in the test set. For multi-frame models, sample 1 contains frames 1-3. Sample 2 contains frames 2-4. Sample 3 contains frames 3-5. As a result, all three frames from sample 2 leak to the training set.

An alternative approach, assigning samples to N folds in-order would result in N different-behaving models. Their predictions and especially activations would differ too much and affect the results of analysis.

Example of in-order fold assignment:

Training samples in the first fold: 8 9 10 11 12 13 14 15 16 17 18 19 20 21

Test samples in the first fold: 1 2 3 4 5 6 7

To minimise direct data leakage while having all models trained on diverse parts of a dataset, a custom fold assignment was implemented. The dataset is split into equal-sized chunks and a portion from each chunk is assigned to each fold.

Example of custom fold assignment building N folds (N=3 in this example):

1. *Split the samples into equal-sized chunks: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21*
2. *Split each chunk into N parts: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21*
3. *For i-th fold, take the i-th part of each chunk and assign it as the test set, assigning the rest of the chunk as the training set.*
    a. *Test samples in first fold: 1 2 3 8 9 10 15 16 17*
    b. *Training samples in first fold: 4 5 6 7 11 12 13 14 18 19 20 21*

Random folds would leak 96% frames when considering the actual parameters used (dataset size, number of folds, chunk size). The custom folds method reduces it to 9.3%.

3.6 Description of Analysis Methods

The goal of the experiments at hand is to validate whether the speed novel to the model causes OOD effects.

3.6.1 Open-loop Evaluation

Open-loop performance of the four model types was measured with mean absolute error. The absolute difference of predicted and ground truth angles was averaged across all data points from all folds (always using a model that left the given fold as a test set).

Each model type ran the forward-pass twice - once on InD test data and once on the OOD data, resulting in 8 mean absolute error values. The same custom 5-fold cross-validation split method was used as described in the training subsection above.

3.6.2 Closed-loop Evaluation

Closed-loop performance was measured for the same model types as for the open-loop performance analysis. As there is no evaluation done using recorded data, folds were not needed and the entire dataset was used for training. This results in just two models (single-frame and multi-frame) per dataset (fast and slow).

Each model was deployed with InD speed and OOD speed. Two 10-lap trials were conducted for both speeds. The battery was always charged to full between trials. The metric used was the number of interventions. An intervention was necessary when the car hit the track wall and could not continue on its own. It was then moved to the start of a new lap.

3.6.3 Multi-frame Inputs Become Out of Distribution

[Haavel] and [Nguyen] showed that when encountering novel types of inputs, the network responds differently than when it encounters known types of inputs. These novel types of inputs are called out-of-distribution (OOD) as they do not fall into the distribution defined by known inputs. In particular, the individual neuron activations are measurably different for known and novel inputs, with the separation becoming even more evident when inputs move farther out of distribution.

In this work, the activations in the second-to-last layer of the multi-frame networks are studied. The activations are from a dense 256-neuron layer after applying ReLu (rectified linear unit activation function), like in [Sun]. The activations were collected while doing a forward pass on the OOD data and on the InD test data, which was not used for model training. Activations in single-frame networks are not studied as we hypothesise that individual frames are not different at different speeds, the difference arises only from considering multiple consecutive frames.

There are multiple ways to detect if a data sample is out of the training data distribution [Bulusu]. One way is to look at the values of the activations (outputs of activation functions) that occur in a model. It has been shown by [Sun] that novel data causes greater positively skewed activation values compared to known data for some models and datasets. Based on this, basic statistics, namely mean, standard deviation and skewness can be evaluated.

The Mahalanobis distance is the *de facto* standard in OOD detection [Denouden]. The approach by [Denouden] assumes that OOD data is composed of different factors than InD data. Therefore, it is difficult to compress and reconstruct OOD data based on a reconstruction scheme optimised for in-distribution data. Denouden incorporated the Mahalanobis distance in latent space to better capture these OOD samples and calculated the Mahalanobis distance between the encoded test sample and the mean vector of the encoded training set.

Computing distance of activation patterns to InD data activation patterns is a typical method for detecting OOD inputs. However, the activations depend not only on whether the input is OOD but also on the type of input and the expected output. As a result, in a classification task, one finds the class centres using training data activations, measures how close the new data point's activation to any of these classes is, and takes the minimum distance from these measurements. Based on thresholding this minimal distance, a decision can be made if the sample is OOD.

Since predicting the turning angle of a car is a regression task, it is unclear how to divide the data into classes to compute multiple "class centres" and the minimum distance. Therefore K-means clustering was used to find cluster centroids. The optimal number of clusters was found empirically, using the elbow method.

The first 9500 samples of InD activations were used in K-means clustering as the training set to find 3 cluster centres, each with coordinates of 256-dimensions. The second part of InD activations was used as a test set to find Mahalanobis distances. Mahalanobis distances were also found for OOD activations.

The Mahalanobis distance to the closest cluster centre was calculated for each sample in the InD test set and in the OOD set. The ROC (Receiver Operating Characteristic) curve was drawn and AUROC (Area Under the ROC curve) was calculated to measure how accurately InD and OOD can be separated based on the Mahalanobis distance.

T-SNE [Belkina] was used to create 3-dimensional embeddings from the 256-dimensional neuron activations. These embeddings were split into test and training data and used training data to train a logistic regression model to distinguish between OOD and InD datasets.

4 Results

This section details the results of the experiments described in the Methodology section. The plots and notebooks used in this section are stored in the publicly available project GitHub repository [Roosild].

4.1 Validating Assumptions About Data

In this preliminary analysis chapter the difference of the two training datasets (fast and slow) are demonstrated to validate our underlying assumptions - we demonstrate the differences in driving speed of our datasets, we show that consecutive frame pixel values differ more at high speed and show that turning angle (the ground truth) distribution depends on speed.

4.1.1 Driving Speed Differences

After cleaning the dataset, the number of frames per lap was counted manually to ensure a relevant speed difference between the two datasets. The frame number at the beginning of each lap was noted down so that the average frame count per lap could be calculated. The speed difference of the two sets is demonstrated on Figure 9.

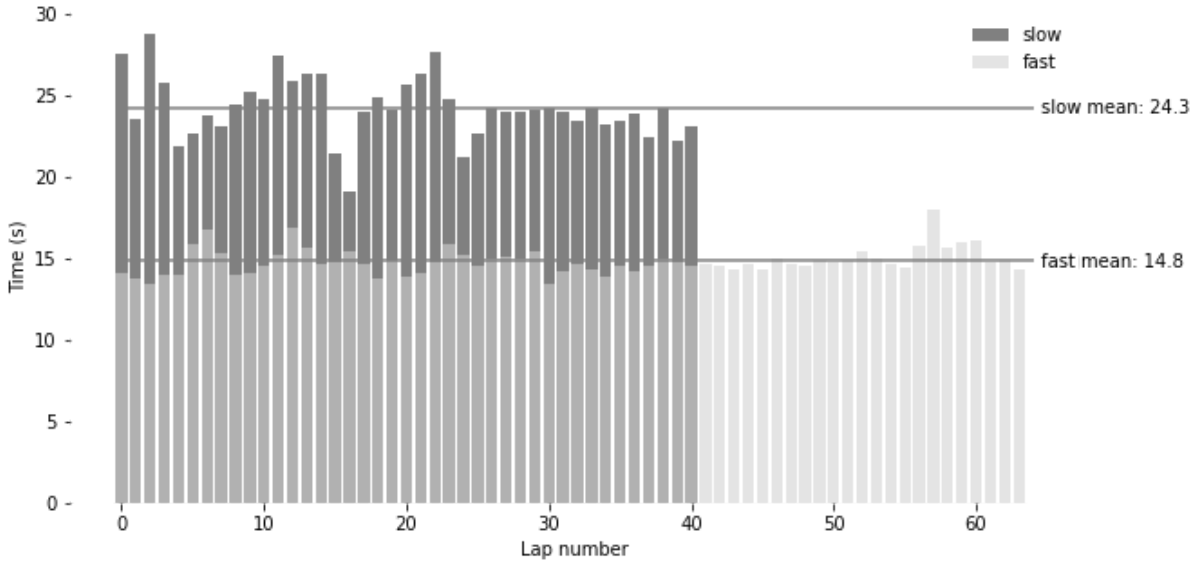


Figure 9. Lap speeds for fast and slow data.

Fast data has 64 complete laps with an average lap time of 14,8 seconds. The standard deviation is 0.8s. Slow data has 41 complete laps with an average lap time of 24,3 seconds. The standard deviation is 1.9s. Datasets' difference in speed is statistically significant ($P < 0.0001$).

Slow data has a bigger standard deviation because it was more difficult to guarantee a constant speed while driving. The ESC heated up faster while driving slower because of the reasons mentioned in the Methodology: Constant Speed subsection.

4.1.2 Frame Differences

We assume multi-frame models suffer when driving at a novel speed partly because frame differences are of a different scale and hence model inputs are of a novel type. Here we validate this assumption that higher speed results in a higher difference between consecutive frames. The average change of the values of RGB pixels in two consecutive images was compared between fast and slow data.

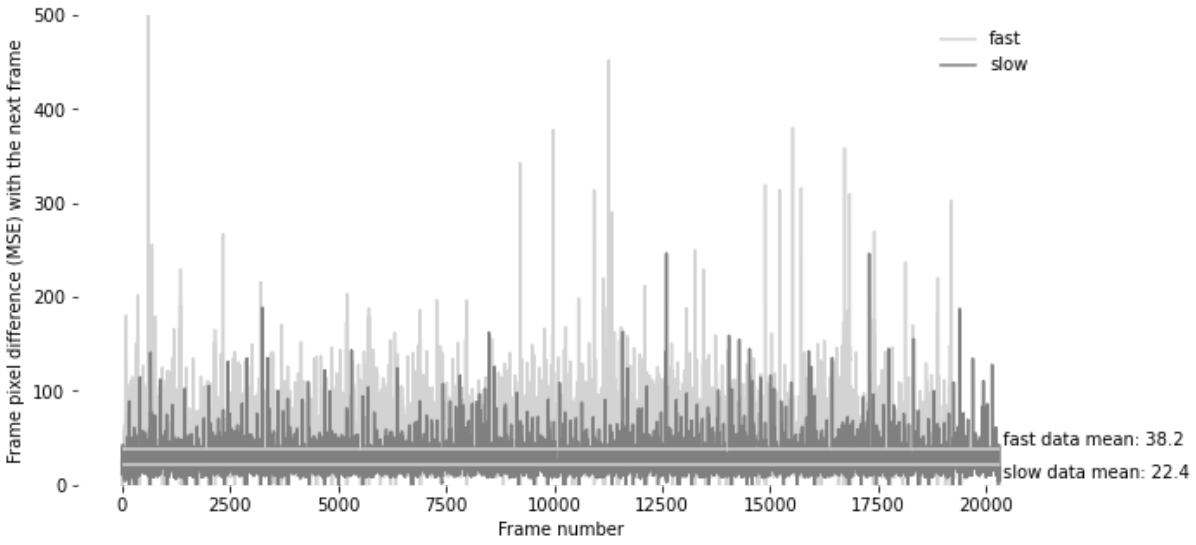


Figure 10. Fast and slow driving frame pixel value differences.

Figure 10 shows that fast data has a more significant pixel-value difference for consecutive frames. This is expected - with faster speed, the car covers more distance between two consecutive frames, thus the environment surrounding the car has changed more during that time.

4.1.3 Ground Truth Turning Angle Distribution

With a faster speed, the centrifugal acceleration combined with inertia results in a higher forward-direction force which the front wheels need to overcome with an even higher force dragging the car to the side. So in order to make a successful turn with the higher speed, the driver needed to steer at a steeper angle. Figure 11 also shows that there are more left angle turns. It is because there was only one right turn on the track and the car was driven

counter-clockwise.

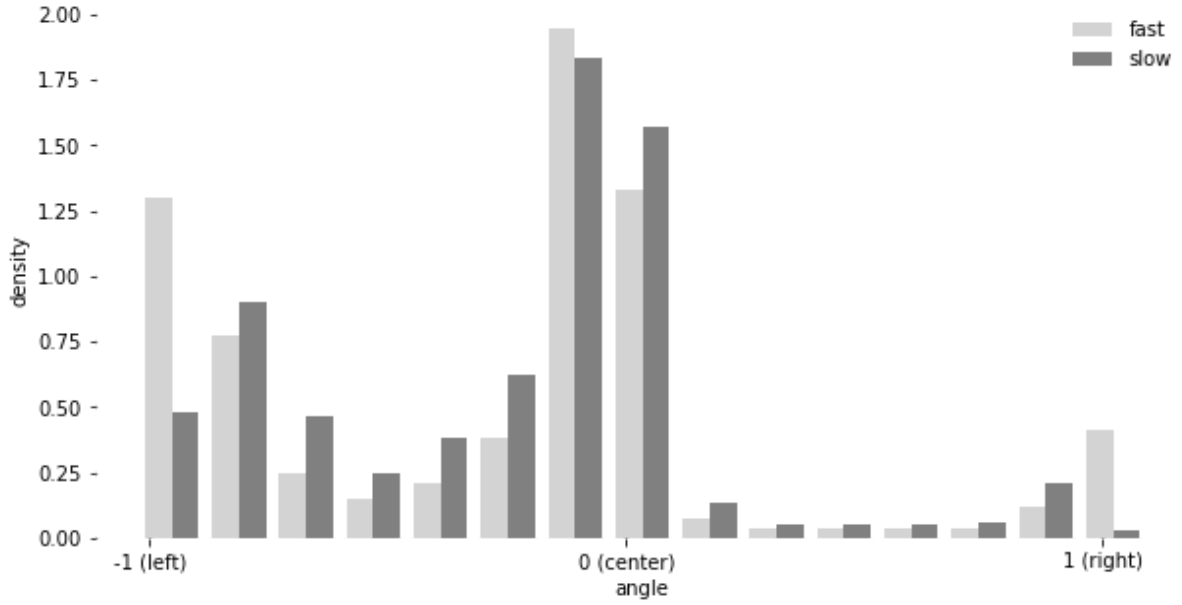


Figure 11. Fast and slow data ground truth angle distribution. On average, the steering angles are more extreme when driving faster. The light grey bars (faster driving) on the diagram are higher on the far left and far right side, while on average, the dark grey bars are higher elsewhere.

4.2 Open-loop Evaluation

Figures 12 and 13 visualise the error of the multi-frame model between ground truth and prediction.

The model which has been trained on fast data performs the same on fast and slow data (Figure 12). The model which has been trained on slow data, performs better with slow data (Figure 13). It seems to predict less extreme angles and thus has a much more significant difference with fast data ground truths (Figure 13, first diagram) than the fast model's predictions have with slow data ground truths (Figure 12, second diagram).

Thanks to the single right turn on the track, the laps are identifiable - the repeating peak of ground truth at around 1.0 angle value (right turn) appears every lap. The slow data during the first 2000 frames has only four laps, while fast data has seven.

As an interesting side note, those diagrams also show that for at least for the first 2000 frames, both datasets have been gathered while driving flawlessly. Both, fast and slow data has one right turn for each five left turns. This 1/5 ratio (one right turn to five left turns) would not exist if some frames would have been cut during the cleaning phase.

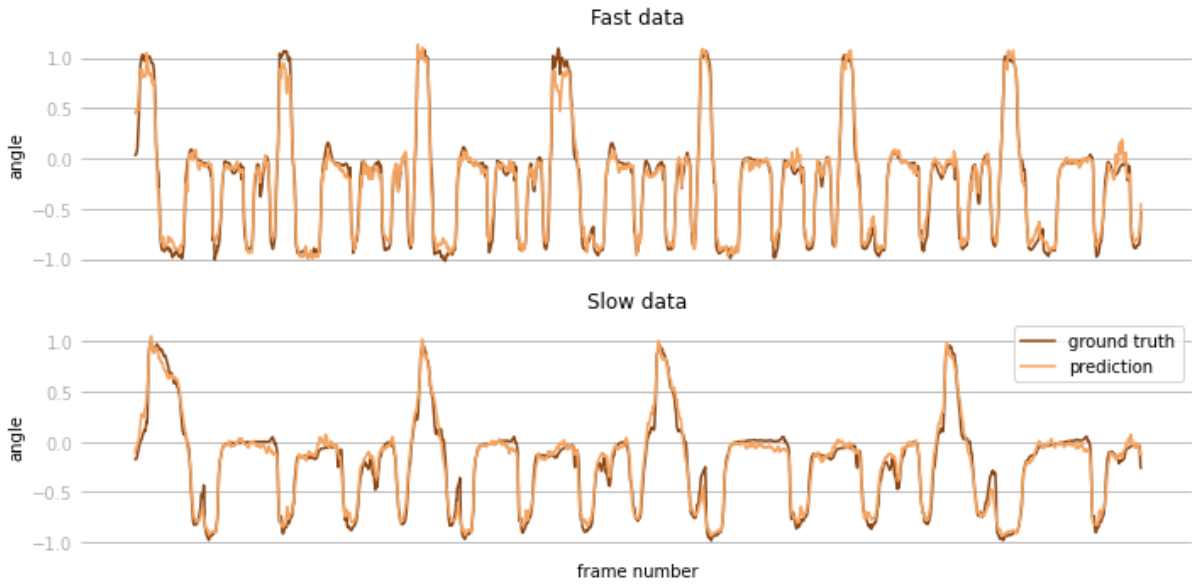


Figure 12. Comparison of ground truth and predicted angles of a multi-frame model trained on fast data.

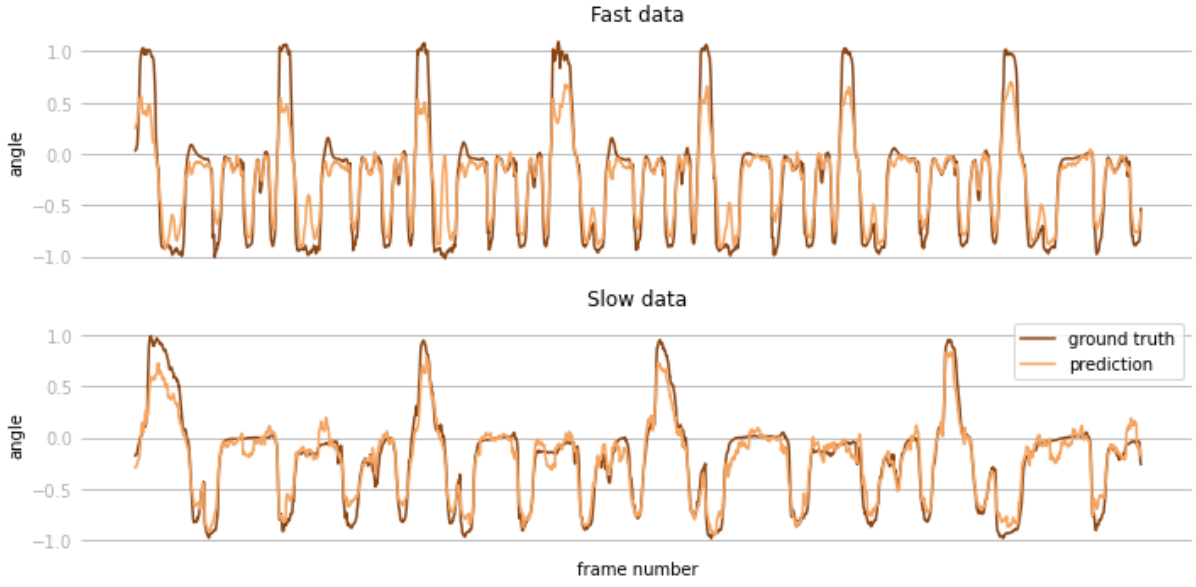


Figure 13. Comparison of ground truth and predicted angles of a multi-frame model trained on slow data.

Tables 1 and 2 show that single and multi-frame models trained on slow data perform worse on fast data, while models trained on fast data only perform a bit worse on slow data than fast data.

This difference in the performance on novel data can be attributed to two findings already depicted before. First, the angles predicted by the model trained on slow data are less extreme than ground truth angles (Figure 13). Second, the slow data ground truth angles are less extreme than fast data ground truth values (Figure 11). Thus, due to these biases, it is easier for our models to make predictions for the slow data.

Therefore, it is not surprising that a model trained on fast data performs almost as well on slow data (multi-frame model's MAE=0.0614) as on fast data (multi-frame model's MAE= 0.0612).

Table 1. Single-frame model open-loop performance.

training data speed	**test data speed**	**mean absolute error**
fast	fast	0.0237
	slow	0.0266
slow	slow	0.0232
	fast	0.0473

Table 2. Multi-frame model open-loop performance.

training data speed	test data speed	mean absolute error
fast	fast	0.0612
	slow	0.0614
slow	slow	0.0888
	fast	0.1298

For both, single- and multi-frame model types, predictions for slow novel test data are more accurate than predictions for fast novel test data. This fact of slow data being easier for all models may hide the effect of OOD for the fast model. Therefore, the average of errors on known vs novel speeds is provided in Table 3.

Table 3. Average open-loop performance of both model types on known and novel speed.

model type	speed	mean absolute error
single-frame	known	0.0235
	novel	0.0367
multi-frame	known	0.0754
	novel	0.0947

As was expected, on average the models perform better on known speed.

4.3 Closed-loop evaluation

It is excellent that open-loop evaluation confirmed our hypothesis but good open-loop performance does not mean good driving ability [Codevilla]. Actual driving ability needs to be measured. A closed-loop evaluation was performed by counting interventions where the car was unable to continue on its own and had to be reset back to the start of the lap.

In Table 4, the number of interventions for single-frame models driving demonstrate that they perform worse with OOD speed.

Table 4. Single-frame model closed-loop performance for two 10-lap trials.

training data speed	**driving speed**	**interventions**
fast	fast	2
	slow (OOD)	16
slow	slow	0
	fast (OOD)	10

The distribution shift of the input (the difference of consecutive camera images) does not affect the single-frame model's predictions. Therefore, the single-frame model performs poorly while driving with OOD speed because of output's distribution shift. In other words, it is the "slowness" of the model which causes the model trained on slow data to turn too late and crash into the outer wall of the turn (Figure 14, blue trajectory) when driving with fast speed. On average, slow data has a smaller absolute turning angle (ground truth). The model turns the car's front wheels only by a little, while it should turn them much more. It takes some time for the model to compute the angle. The servo motor also takes some time to move the wheels to a required angle. There is also some slack between the cogs and levers moving the wheels. All of that amounts to the slow model having learned to turn later and predict smaller absolute angle values than required, causing the car to behave sluggishly and crash outside the turn when driving fast.

A similar explanation applies to the fast single-frame model's bad performance while driving slowly. During our testing, the model turned too early, causing the car to crash into a corner inside the turn (Figure 14, red trajectory).

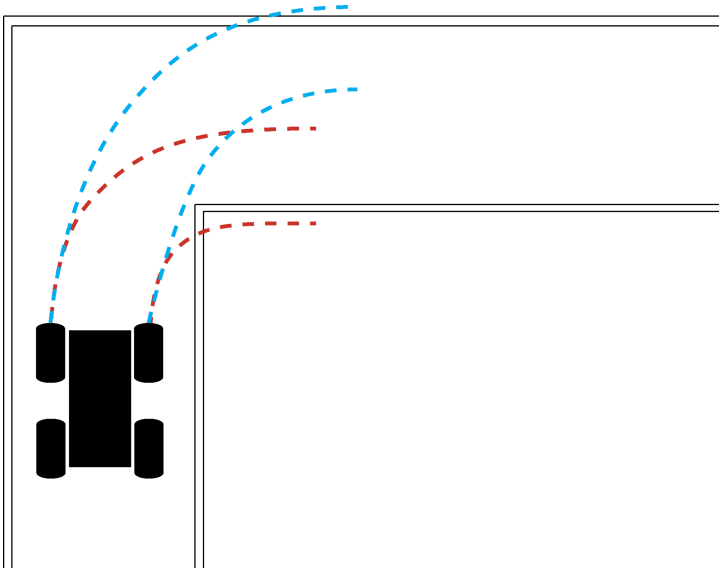


Figure 14. Blue: slow model driving fast. Red: fast model driving slow.

The multi-frame models' closed-loop performance on Table 5 shows similar results to the single-frame models' closed-loop performances - the models perform worse with OOD speed.

Table 5. Multi-frame model closed-loop performance for two 10-lap trials.

training data speed	**driving speed**	**number of interventions**
fast	fast	8
	slow	19
slow	slow	0
	fast	20

It is apparent that similar to open-loop performance, closed-loop testing shows that models perform better at their native speed and worse at an OOD driving speed. For all models, the difference in performance between in and out-of-distribution deployments is statistically significant (p-value > 0.001), as measured by the Binomial test.

We can assume that multi-frame models suffer at OOD speeds due to the output distribution shift that was also present in single-frame models. It is not yet apparent, however, if and how much the multi-frame models' performance suffers from the input being OOD as the analysis performed so far does not separate the input OOD from output OOD.

4.4 Multi-frame Inputs Become Out of Distribution

4.4.1 Activation Skewness

Mean activations have greater variation and strong positive values (i.e., more positively skewed) across units when data is OOD [Sun]. In this subsection, the activation means with standard deviations together with the skewness of mean activation values in the layer of interest are presented.

Table 6 below provides weak evidence that InD and OOD activations behave differently. The average of neuron activation varies more for OOD data (column 4) and also mean skewness is greater for OOD data (column 6).

Table 6. Basic statistics for activations.

model InD speed	data type	mean	standard deviation of neuron means	mean of neuron standard deviations	mean of frame skewness	standard deviation of frame skewness
fast	IND	0.2661	0.0694	**0.3530**	1.5635	**0.8455**
	OOD	0.2837	0.2639	0.2954	1.6707	0.4483
slow	IND	0.3042	0.1102	**0.3867**	1.4697	0.5340
	OOD	0.2073	0.1222	0.2462	1.5382	0.7581

On average, InD data activations have greater standard deviation (0.3530 and 0.3867) and the frame skewness of fast model's InD activations are almost double (0.8455) compared to OOD. This creates doubt, if skewness is a good way of distinguishing between InD and OOD data. The ROC (Receiver Operating Characteristic) curve and AUROC (Area Under the ROC curve) on Figure 15 below shows that skewness cannot be relied on to detect OOD data.

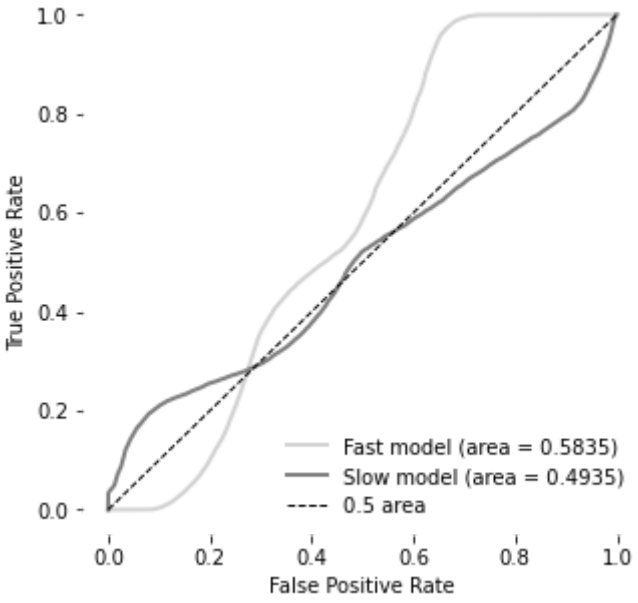


Figure 15. Frame skewness AUROC

4.4.2 Mahalanobis Distance

Summarising the behaviour of a model with mean, standard deviation and skewness values lacks the descriptive capability. Even with the same global statistics, the difference between InD and OOD activation patterns may still exist. For example, different sets of neurons may activate while the global statistics values stay similar.

The Mahalanobis distance is the de facto standard in OOD detection because it correctly measures distances between samples with covarying variables. It does this by first re-scaling variables to remove covariance and then calculates Euclidean distance between the re-scaled points.

We wish to describe activations resulting from in-distribution data via a few cluster centres and then compute minimal mahalanobis distance to these centres as a measure of similarity to training data, i.e. measure of in-distributionness. Elbow method was used to find the number of clusters (N=3) for the K-means clustering method (Figure 16).

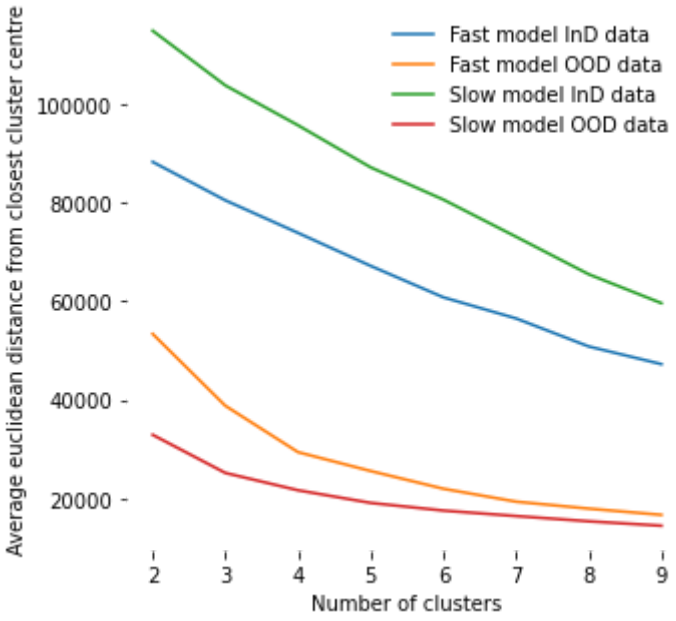


Figure 16. Finding the best number of clusters for K-means clustering method

Cluster fitting and test data were kept separate; the activations for finding the three cluster centres were not used to evaluate the possibility to separate activations of InD and OOD data.

On Figure 17, all four datasets (fast model's activations with InD data, fast model's activations with OOD data, slow model's activations with InD data and slow model's actications with OOD data) are used to derive the ROC curve. Mahalanobis distance score allows for a clear discirmination of InD and OOD data because AUROC is almost 1.0 (0.9998). Appendix I Figure 20 demonstrates an additional method to show that OOD activations differ from InD activations.

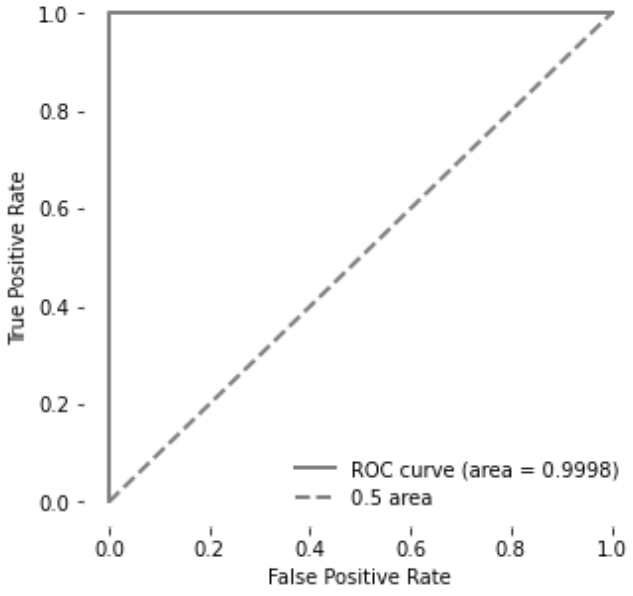


Figure 17. ROC curve for InD and OOD Mahalanobis distances from K-means cluster centres.

Average Mahalanobis distance scores for fast model:

- Train InD 15.6
- Test InD 16.1
- OOD 100.4

Average Mahalanobis distance scores for slow model:

- Train InD 15.7
- Test InD 16.6
- OOD 65.9

This finding shows that the inputs originating from a novel driving speed creates distinguishably different activation patterns in the neural networks hidden layers compared to known speed inputs. It is unlikely that the network is equally optimal to operate in this novel part of activation space it does not encounter when driving at the original speed. Hence, we believe that the decrease in open-loop and closed-loop performance is not only due to outputs being OOD, but also the change in inputs contributes to it.

4.4.3 T-distributed Stochastic Neighbour Embedding

Three-dimensional embeddings were created from the fast model's InD and OOD 256-dimension activations. The 256 dimensions of 39554 samples were reduced to 3 dimensions. The points on the 3-D scatter plot on Figure 18 are coloured based on their belonging to InD or OOD dataset. It is yet another proof of what the ROC curve for Mahalanobis distances showed - the multi-frame model's second-to-last layer activations distribution is very different when the model comes across novel data.

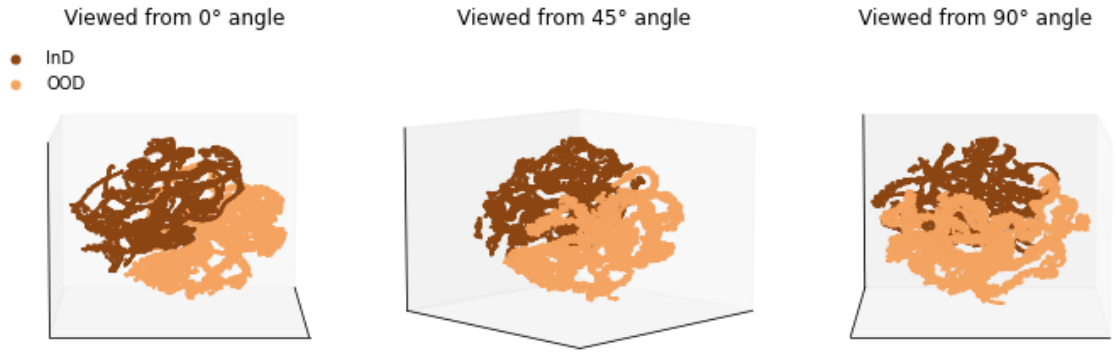


Figure 18. Fast model's t-SNE 3D plot viewed from three different angles.

4.5 Multi-frame Input Distribution Shift Analysis Using Synthesised Data

As an additional analysis, synthetic fast data was created by removing every second frame of the slow data. The resulting data is twice as fast and half the size of the slow data.

The synthesised data has the benefit of being more similar to slow data - both datasets have almost the same distribution of ground truth angles. This reduces the effect of output distribution shift.

There are other other factors which may create an input distribution shift. For example:

- Outdoor lighting from the window, depending on time-of-day, street lamps or headlamps of cars passing by.
- Auto-dimming interior lights in the Delta building.
- Slight changes in driving trajectory, which in part depends on wheel traction which in turn depends on air temperature, humidity and the level of dust on the track.
- Differing standard deviation for lap times, depending on the temperature of ESC and battery charge level.

Since the images in fast synthesised data are all present in the slow data, all the factors listed above are removed from interfering with the driving speed distribution shift analysis results.

The previous findings for multi-frame models were repeated:

- Mean absolute error greater for novel driving speed (known: 0.071, novel: 0.077)
- Skewness is not reliable. Assuming that novel driving speed creates greater positive skewness in the activations, AUROC is only 0.58.
- Mahalanobis distance from Kmeans cluster centres is 17 for known and 66 for novel speed. AUROC is 1.0, clearly separating known and novel speed activations.
- InD and OOD datasets are clearly separable on t-SNE 3-D diagrams, as seen on Figure 19.

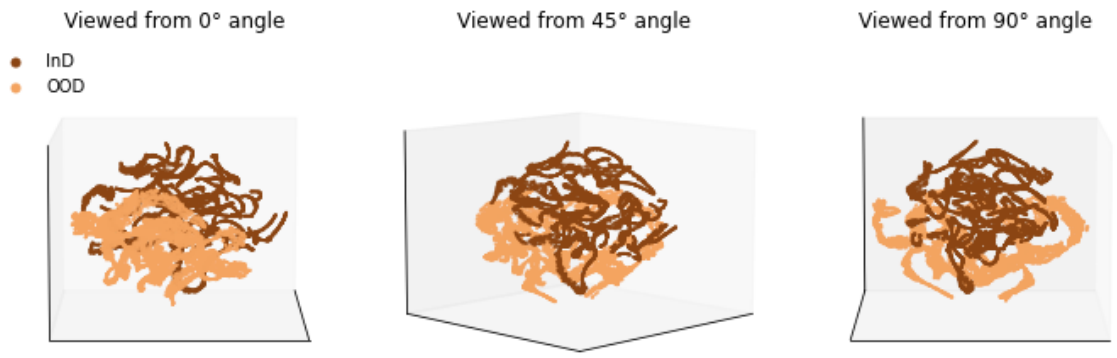


Figure 19. Fast synthesised data model's activations t-SNE 3D plot viewed from three different angles.

After removing a multitude of other factors which may add to the inputs creating a distribution shift, the results with synthesised data on novel driving speed stay unchanged. The novel speed contributes to the decrease in open-loop and closed-loop performance not only because the outputs are OOD but also because the inputs are OOD.

5 Conclusion

The objective of the thesis was to show that the performance of a self-driving system suffers significantly if the deployment speed is novel compared to training speed.

We collected two datasets using the remote-controlled toy car with an onboard camera by driving at a slow and fast speed. We used these datasets to train models with single-frame and multi-frame architecture.

For models with both architecture types, we showed that the open-loop performance is better for the known speed test set compared to the novel speed set. Moreover, we showed similar results for the actual driving performance (closed-loop evaluation) - the car on the track behaved better when the driving speed was known to the model.

The open-loop and closed-loop results could only have been affected by a novel output - a shifted distribution of ground truth turning angles. Hence, using the multi-frame networks' activation layer we demonstrated that novel inputs cause clearly distinguishable activations from known inputs, suggesting that the distribution shift of the inputs is also a factor in worse open-loop and closed-loop performances. The activations resulting from training speed and novel speed data stayed clearly separate in tSNE embeddings and could be almost perfectly separated by a Mahalanobis distance based classifier.

We synthesised fast driving speed data from the slow driving speed data and witnessed the same findings - novel inputs cause clearly distinguishable activations from known inputs.

Future work:

- Analyse if the effect of the novel speed to multi-frame model is reduced by creating synthesised training data with various speeds. The hypothesis is that the model should become more robust and generalise better to different speeds.
- Since there exists more established out-of-distribution detection techniques which consider classification tasks, create a multi-frame multi-class classifier network architecture instead of the regression model. Try to repeat the findings with more of these established techniques.

6 References

[Abdumalikov] R. Abdumalikov and A. Açıkalın.
https://github.com/rabdumalikov/self-driving-donkey-car. Accessed on 01.07.2022.

[Aidla] R. Aidla. Comparing Output Modalities in End-to-End Driving. Msc in preparation, 2022.

[Bansal] M. Bansal, A. Krizhevsky, A. Ogale. ChauffeurNet: Learning to drive by imitating the best and synthesising the worst.

[Belkina] A. Belkina, C. Ciccolella, R. Anno et al. Automated optimized parameters for T-distributed stochastic neighbor embedding improve visualization and analysis of large datasets. Nat Commun 10, 5415, 2019.

[Bojarski] M. Bojarski, D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. Jackel, M. Monfort, U. Muller, J. Zhang, et al. End to End Learning for Self-Driving Cars. arXiv preprint arXiv:1604.07316, 2016.

[Bulusu] S. Bulusu, B Kailkhura, B. Li, P. Varshney, D. Song. Out-of-Distribution Detection in Deep Learning: A Survey.

[Chi] L. Chi and Y. Mu, Deep steering: Learning end-to-end driving model from spatial and temporal visual cues. arXiv preprint arXiv:1708.03798, 2017.

[Codevilla] F. Codevilla, E. Santana, A. Lopez, and A. Gaidon. Exploring the limitations of behavior cloning for autonomous driving. In Proceedings of the IEEE International Conference on Computer Vision, pages 9329–9338, 2019.

[DeltaX] DeltaX Self-driving competition 2022 January.
https://courses.cs.ut.ee/t/DeltaXSelfDriving/Main/2022. Accessed on 01.07.2022.

[Denouden] T. Denouden, R. Salay, K. Czarnecki, V. Abdelzad, B. Phan, and S. Vernekar, Improving reconstruction autoencoder out-of-distribution detection with mahalanobis distance. arXiv, vol. abs/1812.02765, 2018.

[DIY Robocars 2022-1] Outdoors race at Warm Springs Raceway.
https://www.meetup.com/diyrobocars/events/286817397/ Accessed on 15.06.2022.

[DIY Robocars 2022-2] Virtual DonkeyCar (and other cars, too) Race.
https://www.meetup.com/diyrobocars/events/287676337. Accessed on 6.08.2022.

[Donkey Car] An open source DIY self driving platform for small scale cars.
https://www.donkeycar.com. Accessed on 01.07.2022.

[Dosovitskiy] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. CARLA: An open urban driving simulator. arXiv preprint arXiv:1711.03938, 2017.

[Haan] P. de Haan, D. Jayaraman, S. Levine. Causal Confusion in Imitation Learning.

[Haavel] K.Haavel. Exploring Out-of-Distribution Detection Using Vision Transformers, 2022. https://comserv.cs.ut.ee/ati_thesis/datasheet.php?id=74870&year=2022. Accessed on 01.07.2022.

[Kendall] A. Kendall, J. Hawke, D. Janz, P. Mazur, D. Reda, J-M. Allen, V-D. Lam, A. Bewley, A. Shah. Learning to drive in a day. arXiv preprint arXiv:1807.00412, 2018.

[Kurniawan] H. Kurniawan, N. Muhammad, A. T. (2021). Implementing network lag for keras linear with donkey car. Medium. https://handykurniawan.medium.com/implementing-network-lag-for-keras-linearwith-donkey car-43fd726c2afe. Accessed on 01.07.2022.

[LeCun] Y. LeCun, U. Muller, J. Ben, E. Cosatto, B. Flepp. Off-road obstacle avoidance through end-to-end learning. Advances in Neural Information Processing Systems, pages 739–746, 2005.

[Ma] S. Ma, Y. Liu, G. Tao, W.-C. Lee, and X. Zhang, Nic: Detecting adversarial samples with neural network in-variant checking.

[Maison du libre] Maison du libre. https://nantesmakercampus.fr/maison-du-libre. Accessed on 06.08.2022.

[McAllister] R. McAllister, Y. Gal, A. Kendall, M. Van Der Wilk, A. Shah, R. Cipolla, and A. V. Weller, Concrete problems for autonomous vehicle safety: advantages of bayesian deep learning. International Joint Conferences on Artificial Intelligence, Inc., 2017.

[Montgomery] W. D. Montgomery, R. Mudge, E. L. Groshen, S. Helper, J. P. MacDuffie, and C. Carson. America's workforce and the self-driving future: Realising productivity gains and spurring economic growth, 2018.

[Nguyen] A. Nguyen, J. Yosinski, and J Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. Proceedings of the IEEE conference on computer vision and pattern recognition, pages 427–436, 2015.

[NTS Board] NTS Board. preliminary report highway: Hwy18mh010. National Transportation Safety Board, 2018.

[Pomerleau] D. Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In Advances in neural information processing systems, pages 305–313, 1989.

[Renault] The Robocars autonomous car Grand Prix that animated VivaTech 2022. https://www.renaultgroup.com/en/news-on-air/top-stories-2/the-robocars-autonomous-car-grand-prix-that-animated-vivatech-2022. Accessed on 6.08.2022.

[RoboCar Store] Donkey Car Starter Kit. https://www.robocarstore.com/collections/donkey-car/products/donkey-car-starter-kit. Accessed on 6.08.2022.

[Roosild] K. Roosild. Materials of my Data Science Master's thesis. https://github.com/kristjanr/dat-sci-master-thesis. Accessed on 6.08.2022.

[Ross] S. Ross, G. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In Proceedings of the fourteenth international conference on artificial intelligence and statistics, pages 627–635, 2011.

[Sun] Y. Sun, C. Guo, and Y. Li. ReAct: Out-of-distribution Detection With Rectified Activations.

[Tampuu 2020] A. Tampuu, M. Semikin, N. Muhammad, D. Fishman and T. Matiisen. A Survey of End-to-End Driving: Architectures and Training Methods. In Proceedings of the IEEE Transactions on Neural Networks and Learning Systems, 33(4): pages 1364–1384, 2020.

[Tampuu 2022] A. Tampuu, R. Aidla, J. Gent and T. Matiisen. LiDAR-as-Camera for End-to-End Driving. arXiv preprint arXiv:2206.15170, 2022.

[Uduste] I. Uduste. Effect of Delays/Lag and Fighting it in Self-driving Neural Networks

[Wang] D. Wang, C. Devin, Q. Cai, P. Krähenbühl, and T. Darrell. Monocular plan view networks for autonomous driving.

[Waymo] To our fellow San Franciscans.
https://blog.waymo.com/2022/03/to-our-fellow-san-franciscans.html. Accessed on 15.04.2022.

[Wen] C. Wen, J. Lin, T. Darrell, D. Jayaraman, Y. Gao. Fighting Copycat Agents in Behavioural Cloning from Observation Histories.

[Yurtsever] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda. A survey of autonomous driving: Common practices and emerging technologies. arXiv preprint arXiv:1906.05113, 2019.

[Zengg] W. Zeng, W. Luo, S. Suo, A. Sadat, B. Yang, S. Casas, and R. Urtasun. End-to-end interpretable neural motion planner. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 8660–8669, 2019.

Appendix

I Cluster Probability for Ground Truth Angle Classes

The activations of the multi-frame model trained on fast data were assigned into three clusters using the K-means algorithm. The continuous ground truth angle was assigned into three classes - left, centre and right. Figure 20 below shows the probability distribution of clusters in every angle class. While it is evident that clustering activations will not help in distinguishing the angle class, it is clear that OOD distribution differs greatly from InD test distribution.

Figure 20. Cluster probability for ground truth angle classes.

II Model Architectures

Table 7. Single-frame network architecture.

Name	Type	Shape	Number of parameters
img_in	InputLayer	[(None, 61, 160, 3)]	0
conv2d_1	Conv2D	(None, 29, 78, 24)	1824
dropout_2	Dropout	(None, 29, 78, 24)	0
conv2d_2	Conv2D	(None, 13, 37, 32)	19232
dropout_3	Dropout	(None, 13, 37, 32)	0
conv2d_3	Conv2D	(None, 5, 17, 64)	51264
dropout_4	Dropout	(None, 5, 17, 64)	0
conv2d_4	Conv2D	(None, 3, 15, 64)	36928
dropout_5	Dropout	(None, 3, 15, 64)	0
conv2d_5	Conv2D	(None, 1, 13, 64)	36928
dropout_6	Dropout	(None, 1, 13, 64)	0
flattened	Flatten	(None, 832)	0
dense_1	Dense	(None, 100)	83300
dropout_7	Dropout	(None, 100)	0
dense_2	Dense	(None, 50)	5050
dropout_8	Dropout	(None, 50)	0
n_outputs0	Dense	(None, 1)	51
n_outputs1	Dense	(None, 1)	51

Table 8. Multi-frame network architecture.

Name	Type	Shape	Number of parameters
img_in	InputLayer	[(None, 3, 60, 160, 3)]	0
conv3d_6	Conv3D	(None, 1, 20, 53, 16)	1312
max_pooling3d_4	MaxPooling3D	(None, 1, 10, 26, 16)	0
conv3d_7	Conv3D	(None, 1, 10, 26, 32)	4640
max_pooling3d_5	MaxPooling3D	(None, 1, 5, 13, 32)	0
conv3d_8	Conv3D	(None, 1, 5, 13, 32)	9248
flatten_2	Flatten	(None, 2080)	0
dense_4	Dense	(None, 128)	266368
batch_normalization_4	BatchNormalization	(None, 128)	512
activation_4	Activation	(None, 128)	0
dropout_18	Dropout	(None, 128)	0
dense_5	Dense	(None, 256)	33024
batch_normalization_5	BatchNormalization	(None, 256)	1024
activation_5	Activation	(None, 256)	0
dropout_19	Dropout	(None, 256)	0
outputs	Dense	(None, 1)	257

III Acknowledgements

I would like to thank my supervisor Ardi who was always there to help with his bright ideas or suggestions and had patience to guide me on this journey until the glorious end, off- or on-vacation! I am thankful to my coursemate Ilmar, who responded with positivism and reminded me that we are in this together at my bleakest hour of need. My dearest Ljudmilla, who stayed beside me during this whole time and helped by cheering and feedback - thank you! Thank you, my dear friend Bouya for the good words and feedback! Thank you to all my friends and family who understood and patiently waited while they were "put on hold" until finishing this thesis.

IV Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, Kristjan Roosild,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   Driving Speed as a Hidden Factor Behind Distribution Shift,

   supervised by Ardi Tampuu.
2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.
3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.
4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Kristjan Roosild

08/08/2022