Safer End-to-End Autonomous Driving via Conditional Imitation Learning and Command Augmentation

Renhao Wang[1], **Adam Ścibior**[1,2], **Frank Wood**[1,2,3]

[1]University of British Columbia
[2]Inverted AI
[3]MILA
{renhaow, ascibior, fwood}@cs.ubc.ca

Abstract

The driving process with end-to-end autonomous vehicle controllers trained with imitation learning is typically automatic and black-box, although in practice it is desirable to control the vehicle through high-level commands, such as telling it which way to go at an intersection. In existing work this has been accomplished by utilizing a branched neural architecture, since directly providing the command as an additional input to the controller often results in the command being ignored. In this work we overcome this limitation by learning a disentangled probabilistic latent variable model that generates the steering control. We achieve faithful command-conditional generation without using a branched architecture and demonstrate improved vehicle stability. We also extend our model with an additional latent variable and augment the dataset to train a controller that is robust to unsafe commands, such as asking it to turn into an obstacle.

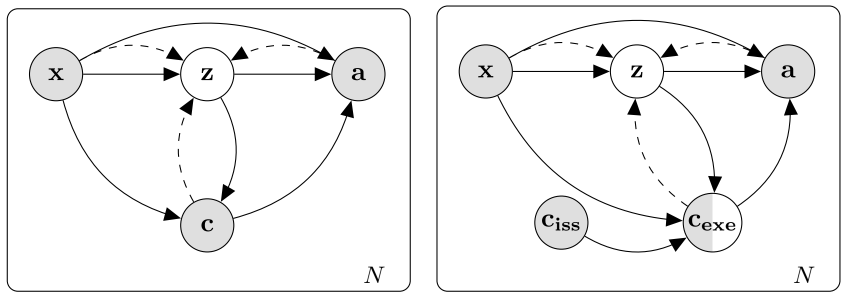
1 Introduction

Traditionally, autonomous driving is done by a software pipeline consisting of perception, localization, planning, and control. However, in recent years an end-to-end approach, mapping perceptual inputs directly to steering actions, has gained popularity. The dominant paradigm for end-to-end training of driving agents is imitation learning (IL) from human demonstration [1, 3, 4, 5, 6, 8, 9]. IL is a promising approach because autonomous vehicles that drive like humans are more predictable and therefore safer. Additionally, it is very easy to obtain data for imitation learning by recording trips taken by human drivers on public roads.

Many initial efforts in IL for autonomous driving were restricted to keeping a vehicle in lane and could not navigate through intersections or accept commands of any kind [3, 6, 9]. To remedy this problem, Codevilla et al. [4] performed end-to-end conditional imitation learning in a simulated environment to train a model capable of behaving as a "chauffeur" and manipulating vehicular dynamics such as steering angle and acceleration according to high level commands.

In this paper we build on the aforementioned work of Codevilla et al., who reported that concatenating the command with the perceptual input and feeding them both into a neural network trained to imitate human behavior often leads to the network ignoring the command at test time. We refer to this non-branched model as DNN. Due to this problem, they employ a branched architecture, effectively training a separate network for each command where the perceptual input is processed by several shared layers before being passed to command-specific layers that output the steering action. We refer

Machine Learning for Autonomous Driving Workshop at the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.



(a) (b)

Figure 1: (a) A probabilistic graphical model describing our command conditional model NDID. (b) The adjusted graphical model (CANDID) for robustness against unspecified or maligned commands. The one-hot command $\mathbf{c}$ is separated into $\mathbf{c_{iss}}$, the driver-issued command, and $\mathbf{c_{exe}}$, the command that the vehicle should execute. $\mathbf{a}$ represents the action, $\mathbf{z}$ represents the latent variable, and $\mathbf{x}$ represents the concatenated featurization of perceptual input and vehicle speed. Shaded circles indicate observed variables, transparent circles for latent variables. $\mathbf{c_{exe}}$ is observed at training time but not at test time. The directed acyclic graph connected by solid arrows denotes the generative probabilistic model; the dashed arrows comprise of the inference network.

to this model as Branched-DNN. Branched-DNN tends to unnecessarily slow down the vehicle pick unnatural trajectories at turns. Inspecting the code provided by Codevilla et al.[1], we found that they use additional hand-coded heuristics to ameliorate this problem. We refer to their model including these heuristics as Branched-Heuristics.

Our first contribution is to replace the branched neural network with a disentangled latent variable model [2], with an interpretable variable corresponding to the command being executed, as depicted in figure 1. We find that this model reliably performs conditional generation when the command latent variable is set to the desired value, and that it produces smoother trajectories than Branched-Heuristics, even without using any hand-crafted heuristics. We refer to our model as the Neural Directable Imitation Driver (NDID).

In a practical application we would also expect the controller to be robust to unsafe commands. In particular, our work concerns unsafe commands such as asking the vehicle to turn when there is no road to turn into, as illustrated in Figure 2, and conversely not telling it to turn at a T-shaped intersection. To achieve such robustness we further extend our probabilistic model with separate variables corresponding to the command being issued and to the command being executed. We train this model on the originally collected dataset, performing automatic data augmentation depicted in Table 2 and described in Section 3. This results in a model robust to unsafe commands, which cause the other models we test to drive off the road but which our model learns to ignore while still obeying safe commands. We call this model the Command Augmented Neural Directable Imitation Driver (CANDID).

2 Latent Variable Model for Command Conditional Driving

Our first contribution involves replacing the branched neural network generating actions in [4] with a probabilistic latent variable model, depicted in Figure 1a. The latent variable is $\mathbf{z}$, which is continuous and represents the uninterpretable "mental state" of the driving agent. The model also includes observable variables $\mathbf{c}$, which is a discrete command, $\mathbf{a}$, which is the action consisting of a steering angle and the amount of force applied to the braking and acceleration pedals, and $\mathbf{x}$, which is the perceptual input consisting of a frame captured by the camera and the vehicle velocity. Note that we do not model the distribution $p(\mathbf{x})$, treating $\mathbf{x}$ as given. All the conditional distributions are parameterized by neural networks, the weights of which are optimized through the standard variational objective called the evidence lower bound (ELBO), defined as:

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},\mathbf{a},\mathbf{c})}\left[\log \frac{p_\theta(\mathbf{z},\mathbf{a},\mathbf{c} \mid \mathbf{x})}{q_\phi(\mathbf{z} \mid \mathbf{x},\mathbf{a},\mathbf{c})}\right] \leq \log p_\theta(\mathbf{a},\mathbf{c} \mid \mathbf{x})$$

[1] https://github.com/carla-simulator/imitation-learning

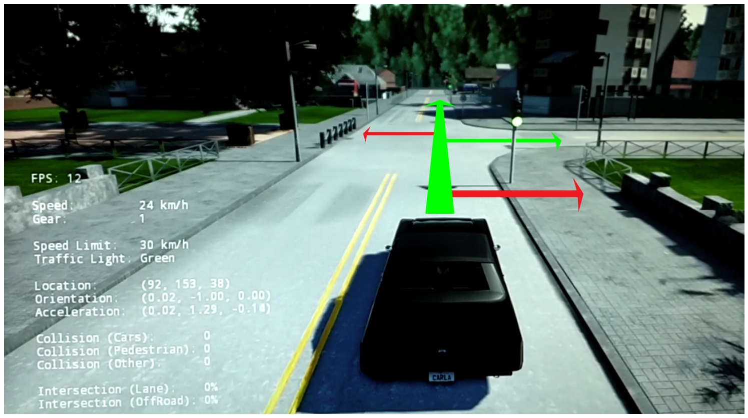


Figure 2: A vehicle is operating autonomously, following commands instructing it which way to take at intersections. We introduce a command augmentation scheme that allows us to train a controller to obey safe commands (in green) while ignoring unsafe commands (in red).

where ϕ and θ parameterize the inference and generative networks, respectively.

This approach to model learning was introduced by Kingma and Welling [7] as a variational autoencoder (VAE), but note that we do not attempt to reconstruct the scene from the latent variables. Our model could be regarded as a conditional VAE [10] of the action given perceptual inputs, but such an interpretation is misleading since the dimension of $\mathbf{z}$ is larger than the dimension of $\mathbf{a}$.

After training, we achieve command conditional generation by setting $\mathbf{c}$ to the desired value, then sampling $\mathbf{z} \sim p(\mathbf{z} \mid \mathbf{x})$ and $\mathbf{a} \sim p(\mathbf{a} \mid \mathbf{c}, \mathbf{z}, \mathbf{x})$. Qualitatively we find that our model NDID not only obeys the commands issued but also executes turns more smoothly and at more human-like speeds than Branched-Heuristics, despite NDID not using any hand-crafted adjustments to generated actions. Quantitatively, we evaluate our model on the benchmark used by Codevilla et al. [4], which consists of a series of driving tasks. Each task is comprised of an initial location where the agent is initialized, and a final destination that the agent must navigate to using high level commands provided by an A* topological planner. We measure both the fraction of these trips where the vehicle is able to reach the destination, as well as the error-free travel distance as driving infractions occur, such as leaving the lane or colliding with fixed obstacles (see Table 1).

Table 1: Performance of our model NDID (Figure 1a) compared against existing models on the navigational benchmark used by Codevilla et al. [4]. All training data was recorded in Town 1. Branched-Heuristics is the full model used by Codevilla et al., Branched-DNN is the same model without the hand-crafted adjustments to generated actions, and DNN is a non-branching neural controller that received the command as input.

Model	Success Rate		Km Between Infraction	
	Town 1	Town 2	Town 1	Town 2
DNN	56%	32%	0.76	0.14
Branched-DNN	73%	54%	1.45	0.93
Branched-Heuristics	84%	61%	2.12	1.08
NDID	**88%**	**71%**	**6.80**	**4.53**

3 Command Augmentation for Increased Safety

A common underlying assumption for all the models presented in the previous section is that the commands are always issued correctly and correspond to actions that make sense. However, in practice a controller may erroneously issue the incorrect command (for example, commanding a left turn where none is possible on a straight stretch of road) or fail to issue a command at all (for example, failing to issue left or right turn commands at a T-intersection).

Table 2: Illustration of our data augmentation scheme facilitating robustness to badly issued commands. (x, a) and (x', a') are arbitrary pairs of sensory inputs and vehicle actions present in the non-augmented dataset. In the non-augmented dataset the "stay in lane" command is only issued outside of intersections where turn commands should be ignored, so in those situations the controller should always execute "stay in lane". Turn commands indicate a presence of an intersection, where "stay in lane" corresponds to a missing command, in which case the controller should take an action known to be safe. The augmentation we use in that case is to change the issued command to "stay in lane" but keep the executed command the same.

Image	Speed	Command
x	a	stay in lane
·	·	·
·	·	·
·	·	·
x'	a'	left turn
·	·	·

$\Rightarrow$

Image	Speed	Command Issued	Command Executed
x	a	stay in lane	stay in lane
x	a	straight	stay in lane
x	a	right turn	stay in lane
x	a	left turn	stay in lane
x'	a'	left turn	left turn
x'	a'	stay in lane	left turn

To address this, we extend our probabilistic model to demarcate from the command issued, $\mathbf{c_{iss}}$, and the command that the vehicle executes, $\mathbf{c_{exe}}$. The corresponding graphical model is presented in Figure 1*b*. We train the model to recognize whether the command issued is safe to execute, and if not to choose a safe one. We achieve this by augmenting the original dataset without modifying the training procedure. We call the resulting model CANDID.

Concretely, we augment the dataset in two ways to achieve i) robustness against unsafe commands and ii) sensible default driving when no commands are provided, respectively. Our data augmentation scheme is presented in Table 2.

To train this model we again maximize the ELBO, now conditioning on both sensory input $\mathbf{x}$ as well as the issued command $\mathbf{c_{iss}}$.

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},\mathbf{a},\mathbf{c_{exe}})}\left[\log \frac{p_\theta(\mathbf{z},\mathbf{a},\mathbf{c_{exe}} \mid \mathbf{x},\mathbf{c_{iss}})}{q_\phi(\mathbf{z} \mid \mathbf{x},\mathbf{a},\mathbf{c_{exe}})}\right] \leq \log p_\theta(\mathbf{a},\mathbf{c_{exe}} \mid \mathbf{x},)$$

where once again ϕ and θ parameterize the inference and generative networks, respectively.

To evaluate how our model reacts to unsafe commands, we firstly introduce turn commands when the vehicle is proceeding along straight stretches of road where no turns are possible. All the models presented in Section 2 veer off the road, whereas CANDID ignores the unsafe command and proceeds straight ahead. Secondly, we provide the vehicle only with the "stay in lane" command and observe how it behaves at a T-intersection. All the models from Section 2 obey the command and proceed directly into the barrier at the end of the intersection. Our CANDID model trained on the augmented dataset is able to turn right and avoid the crash.

For quantitative evaluation we use the same set of driving tasks we used in Section 2 but randomly change 10% of all commands issued by the simulator to an alternative command (e.g. the vehicle may be told to turn left instead of right at a particular intersection, or to turn when proceeding along a straight stretch of road). The results are presented in Table 3 and show that performance severely degrades for all models except CANDID.

Table 3: Performance of our model CANDID on a benchmark that includes badly issued commands. All the other models presented in this table reliably fail when given a bad command, their success rate being non-zero only due to presence of very short driving tasks.

Model	Success Rate		Km Per Infraction	
	Town 1	Town 2	Town 1	Town 2
DNN	13%	7%	0.46	0.12
Branched-DNN	16%	11%	0.57	0.23
Branched-Heuristics	19%	10%	0.53	0.21
NDID	18%	11%	0.62	0.25
CANDID	**83%**	**65%**	**5.93**	**3.13**

References

[1] Bansal, Mayank and Krizhevsky, Alex and Ogale, Abhijit. Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. *arXiv preprint arXiv:1812.03079*, 2018.

[2] Bengio, Yoshua and Courville, Aaron and Vincent, Pascal. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[3] Bojarski, Mariusz and Del Testa, Davide and Dworakowski, Daniel and Firner, Bernhard and Flepp, Beat and Goyal, Prasoon and Jackel, Lawrence D and Monfort, Mathew and Muller, Urs and Zhang, Jiakai and others. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

[4] Codevilla, Felipe and Miiller, Matthias and López, Antonio and Koltun, Vladlen and Dosovitskiy, Alexey. End-to-end driving via conditional imitation learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–9. IEEE, 2018.

[5] Dosovitskiy, Alexey and Ros, German and Codevilla, Felipe and Lopez, Antonio and Koltun, Vladlen. CARLA: An open urban driving simulator. *arXiv preprint arXiv:1711.03938*, 2017.

[6] Kendall, Alex and Hawke, Jeffrey and Janz, David and Mazur, Przemyslaw and Reda, Daniele and Allen, John-Mark and Lam, Vinh-Dieu and Bewley, Alex and Shah, Amar. Learning to drive in a day. *arXiv preprint arXiv:1807.00412*, 2018.

[7] Kingma, Diederik P and Welling, Max. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[8] Pan, Yunpeng and Cheng, Ching-An and Saigol, Kamil and Lee, Keuntaek and Yan, Xinyan and Theodorou, Evangelos and Boots, Byron. Agile off-road autonomous driving using end-to-end deep imitation learning. *arXiv preprint arXiv:1709.07174*, 2017.

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

[10] Sohn, Kihyuk and Lee, Honglak and Yan, Xinchen. Learning structured output representation using deep conditional generative models. In *Advances in neural information processing systems*, pages 3483–3491, 2015.