Adversarial Mixture Density Networks: Learning to Drive Safely from Collision Data

Sampo Kuutti, Saber Fallah, Richard Bowden

Abstract— Imitation learning has been widely used to learn control policies for autonomous driving based on pre-recorded data. However, imitation learning based policies have been shown to be susceptible to compounding errors when encountering states outside of the training distribution. Further, these agents have been demonstrated to be easily exploitable by adversarial road users aiming to create collisions. To overcome these shortcomings, we introduce Adversarial Mixture Density Networks (AMDN), which learns two distributions from separate datasets. The first is a distribution of safe actions learned from a dataset of naturalistic human driving. The second is a distribution representing unsafe actions likely to lead to collision, learned from a dataset of collisions. During training, we leverage these two distributions to provide an additional loss based on the similarity of the two distributions. By penalising the safe action distribution based on its similarity to the unsafe action distribution when training on the collision dataset, a more robust and safe control policy is obtained. We demonstrate the proposed AMDN approach in a vehicle following use-case, and evaluate under naturalistic and adversarial testing environments. We show that despite its simplicity, AMDN provides significant benefits for the safety of the learned control policy, when compared to pure imitation learning or standard mixture density network approaches.

I. INTRODUCTION

Autonomous vehicles have received increasing interest as a potential solution to transportation issues such as traffic congestion, pollution, and vehicle safety [1]–[3]. One of the enabling technologies gaining popularity for autonomous driving is deep learning [4]. Deep learning has been demonstrated to learn driving rules from recorded data, which provide general driving policies that generalise to a wide variety of driving scenarios and outperform traditional approaches.

A common strategy for learning autonomous driving policies is imitation learning, where a deep neural network is trained to imitate an expert driver using a dataset of demonstrated driving [5]. Imitation learning has shown to scale well to large amounts of driving data, and to perform well when deployed in similar scenarios to its training data [6]–[9]. However, one of the main limitations of imitation learning is that during training, the states observed by the agent are not dependent on the learned control policy. However, once deployed, the states visited by the agent depend on its own control policy, often leading to compounding errors as the agent visits states outside of its training distribution [10]–[13]. As the agent experiences events further away from the

safe states seen in its training data, the errors in its prediction grow larger, creating a cascading failure. Further issues in imitation learning models can be caused by unpredictable road users [14] or causal confusion [15].

A number of solutions have been offered to mitigate these drawbacks in imitation learning. For instance, Dataset Aggregation (DAgger) [13], first learns from a dataset of expert demonstration after which the agent is deployed in the test environment. During testing, the expert labels each state seen by the agent with the correct action, and in this way, labels for states visited by the agent not seen in the original training data are collected. However, one of the main limitations of this approach is that it requires the expert to be available for labelling large amounts of new data, which is not always possible. Moreover, for a human annotator it may be difficult to label the data when they were not in control of the vehicle (e.g. labelling the correct control action may be difficult for a human when presented with only a still image frame, without further context). An alternative approach that does not query the expert during training is Generative Adversarial Imitation Learning (GAIL) [16], inspired by Generative Adversarial Networks [17]. GAIL uses a discriminator during training, which aims to classify trajectories as either coming from the expert or the learned model. Therefore, as training progresses the trajectories by the learned model become more human-like, making the classification more difficult for the discriminator. Other approaches have turned to Reinforcement Learning (RL) instead, which learns from interaction with its environment [18]–[21]. As the states visited by the RL agent during training are dependent on its own policy, RL tends to provide a more robust policy which can generalise well to different scenarios. However, RL requires a significant amount of training, due to the low sample efficiency of RL algorithms [22]. This typically leads to the need for high fidelity simulators, and restricts the use of RL within real-world use-cases.

In this work, we present Adversarial Mixture Density Networks (AMDN) which provide a method for imitation learning with improved robustness to distribution shift and increased safety in dangerous driving scenarios. The proposed techniques utilises two datasets, one with safe expert driving trajectories with labelled actions and another with trajectories that lead to collisions without labels for correct actions. The AMDN model aims to simultaneously learn two distributions representing both safe and unsafe actions. By learning these two distributions, the agent can then penalise the safe action distribution whenever it becomes too similar to the unsafe action distribution. Using these two distribu-

Sampo Kuutti and Saber Fallah are with the Connected and Autonomous Vehicles Lab, University of Surrey, Guildford, GU2 7XH, UK. Email: {s.j.kuutti, s.fallah}@surrey.ac.uk

Richard Bowden is with the Centre for Vision, Speech and Signal Processing, University of Surrey, GU2 7XH, UK. Email: r.bowden@surrey.ac.uk

tions, and a measure of similarity between them, the model can then further train the safe action distribution based on the safety-critical scenarios in the collision dataset, without any need for labelling of this dataset. This is similar to Linear Discriminant Analysis (LDA) [23], [24], which aims to separate classes of data by modelling them as Gaussian distributions, and maximising the mean distance between each class [25]. However, our approach differs significantly from LDA, since in our case the aim is to use regression to infer continuous control actions rather than classify data or reduce the dimensionality of the data. And since the aim is to learn a safer control policy, we only penalise the safe action distribution based on its similarity to the unsafe distribution. Furthermore, by leveraging the non-linearity of deep neural networks, the proposed technique can be trained on more complex data and use-cases. Therefore, our proposed technique extends standard imitation learning and mixture density network approaches, by providing an additional loss during imitation learning, which increases the safety of the learned driving policy and boosts the model's robustness to adversarial agents. We demonstrate the proposed approach in a vehicle following use-case, and evaluate the control policies both in naturalistic driving with typical leading vehicles, as well as in safety-critical environments with adversarial agents who are attempting to create rear-end collisions. We demonstrate that the proposed Adversarial Mixture Density Networks improves the model's robustness in these safety-critical scenarios compared to normal imitation learning and mixture density networks.

The remainder of the paper is as follows. Section II presents the required background and introduces the proposed Adversarial Mixture Density Network technique. Section III demonstrates the effectiveness of the approach in simulated driving tests. Finally, conclusions are presented in Section IV.

II. METHODOLOGY

We demonstrate the proposed approach in a highway vehicle following use-case. The aim of the model is to control the longitudinal actions of an autonomous vehicle, whilst following the lead vehicle at a safe distance. The model does this by observing states from the vehicle's Radar and inertial sensors, and then infers the correct actions to control the gas and brake pedals of the host vehicle. We give a brief overview of the background on Imitation Learning and Mixture Density Networks in sub-sections II-A and II-B, respectively. We then introduce the proposed Adversarial Mixture Density Networks in sub-section II-C.

A. Imitation Learning

Imitation Learning (IL) is a type of supervised learning strategy, where an agent learns from expert demonstration of the desired task. The aim is to learn to imitate the expert's behaviour [26]. To do this, an expert demonstration dataset $\mathcal{D}^e$, consisting of observed states s_t and the corresponding actions taken by the expert $\hat{a}_t$, is used. The IL policy denoted by π^{IL} and represented by parameters θ^{IL}, aims to find

optimal parameters θ^*, by minimising an imitation loss $\mathcal{L}^{IL}$ based on the distance of its predicted action a_t to the expert's demonstrated action $\hat{a}_t$ for the same observed state s_t:

$$\theta^* = \arg\min_{\theta^{IL}} \sum_t \mathcal{L}^{IL}(\pi^{IL}(s_t^{IL}|\theta^{IL}), \hat{a}_t) \qquad (1)$$

B. Mixture Density Networks

A Mixture Density Network (MDN) [27] is a type of neural network, which can model a distribution as a mixture of parametric distributions:

$$p(y|x) = \sum_{i=1}^{M} \alpha_i(x)\phi(y|\theta_i) \qquad (2)$$

where y is the output, x is the input, M is the number of mixture components, α_i is the mixing coefficient, and ϕ is a parametric distribution with parameters θ_i. For instance, a Gaussian distribution $\mathcal{N}(y|x)$ can be represented by its mean and (co-)variance $\theta_i = (\mu_i, \sigma_i^2)$.

Mixture Density Networks can be used to estimate any given density function to an arbitrary accuracy given sufficient number of mixtures [28]. MDNs have been used to model data distributions thanks to their ability to represent multi-modal data distributions and provide uncertainty estimations in various tasks such as speech synthesis [29], future prediction [30], and autonomous driving [31], [32].

C. Adversarial Mixture Density Networks

Our proposed Adversarial Mixture Density Network is similar to MDNs in that it also aims to represent parametric distributions given an input x. However, instead of learning a mixture of distributions that maps the probability of output y for a given x, the AMDN aims to learn two different (and often opposite) distributions $p(y|x)$ and $p(\bar{y}|x)$ for the same input x. For the vehicle following use-case presented here, these distributions represent the *safe* action a^s in current state s and the *unsafe* action a^c in the same state. Simply, the safe action a^s is learned from an expert demonstration of the task, whilst the unsafe action is learned from a dataset of collisions and represents an action likely to lead to a collision in the short term. Therefore, for any given state s_t, the AMDN models the probability of safe action $p(a_t^s|s_t)$ and the probability of action leading to collisions $p(a_t^c|s_t)$. For the vehicle following use-case presented here, unimodal policies are sufficient to solve the task, and since we are not using the different distributions to estimate uncertainty [32] or multi-modal policies [33], we represent each distribution by a unimodal Gaussian distribution, such that $p(a_t^s|s_t) = \mathcal{N}^s(\mu^s, \sigma^{s2})$ and $p(a_t^c|s_t) = \mathcal{N}^c(\mu^c, \sigma^{c2})$. However, it is worth noting that it would be straightforward to extend our approach to use mixtures of density functions for each distribution, should multi-modal representations be desired for the given task.

This additional information about safe and unsafe actions can then be employed to learn a safer control policy, by penalising the safe action distribution when it becomes similar to the unsafe action distribution. To do this, a similarity measure between the two distributions is required.
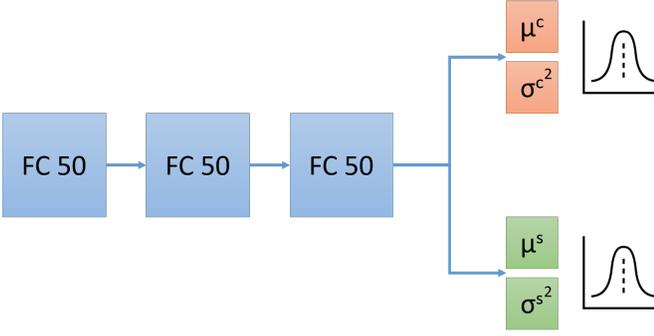
Fig. 1: Proposed Adversarial Mixture Density Network architecture.

TABLE I: AMDN hyperparameters.

Parameter	Value
Safe action learning rate, η_s	1x10^{-4}
Unsafe action learning rate, η_c	1x10^{-5}
KL-divergence learning rate, η_{KL}	1x10^{-9}
Hidden layers	3
Hidden neurons per layer	50
Batch size	100
Training steps	1x10^6

A common statistical measure of the difference between two distributions is the Kullback-Leibler (KL) divergence D_{KL} [34]. For two distributions the KL-divergence is given as:

$$D_{KL}(p(a_t^s|s_t)||p(a_t^c|s_t)) = \sum p(a_t^s|s_t)log\left(\frac{p(a_t^s|s_t)}{p(a_t^c|s_t)}\right) \tag{3}$$

The AMDN is represented by a feedforward neural network, with three hidden layers of 50 neurons each, followed by the output layers which represent the mean and variance of the two Gaussian distributions μ^s, σ^{s2}, μ^c, and σ^{c2}, as shown in Fig. 1. The two distributions are each trained on their respective datasets, $\mathcal{D}^e = (s_t^s, \hat{a}_t^s)$ and $\mathcal{D}^c = (s_t^c, \hat{a}_t^c)$. Both datasets are split into training and validation splits by a 80%/20% ratio, while further closed-loop testing within the simulator is used to test the robustness of the final trained policies. For both datasets, the action of the agent a_t is represented by a single value in the range [-1, 1], which represents the use of the gas and brake pedals of the host vehicle, where positive values indicate the use of the gas pedal and negative values the brake pedal. The observed states s_t consist of the host vehicle velocity v, relative velocity to the lead vehicle v_{rel}, and time headway to the lead vehicle t_h. The time headway is a measure of inter-vehicle distance in time, given by:

$$t_h = \frac{x_{rel}}{v} \tag{4}$$

where x_{rel} is the relative distance between two vehicles in m, and v is the velocity of the host vehicle in m/s.

The expert dataset $\mathcal{D}^e$ is the dataset presented in [35], which consists of 2 hours of driving data by the IPG Car-Maker Simulator's [36] default driver (IPG Driver) driving in different highway driving scenarios while aiming to maintain a time headway of 2s. The dataset consists of a total of 375,000 observation-action pairs.

The collision dataset $\mathcal{D}^c$ is a dataset collected by utilising the adversarial testing framework in [14] and stress testing a vehicle following controller. The testing framework uses adversarial agents to evaluate the target control policy. The adversarial agents are trained through reinforcement learning, with the aim of learning behaviours which cause the follower vehicle to crash into it. To ensure the collisions are preventable, and therefore expose a weakness in the target control policy, the actions and states of the adversaries are constrained based on minimum/maximum velocities and accelerations. The reward function of the adversary $r_t^A(s_t^A, a_t^A)$ is based on the inverse time headway to the follower vehicle, given by

$$r_t^A = min(\frac{1}{t_h}, 100) \tag{5}$$

Where r_t^A is the adversary's reward at time-step t, and the reward is capped at 100 to avoid the reward function tending towards infinity as the headway reaches 0. To collect the dataset, the adversary was deployed in different velocity ranges in highway driving, and the follower vehicle was controlled by an imitation learning agent trained on the dataset $\mathcal{D}^e$. In this way, a dataset representing over 11,000 collision scenarios was collected. For each collision in the dataset, the 25 time-steps before the collision are taken for a total of 1s of driving data per collision and used for training, for a total of 275,000 observation-action in the dataset.

We train the two distributions $\mathcal{N}^s$ and $\mathcal{N}^c$ with a Negative Log-Likelihood (NLL) loss, on the $\mathcal{D}^e$ and $\mathcal{D}^c$ datasets, respectively. The NLL loss $\mathcal{L}_{NLL}$ maximises the likelihood of given label actions $\hat{a}_t$ given an observation s_t:

$$\mathcal{L}_{NLL}(s_t, \hat{a}_t) = -log(\mathcal{N}(\hat{a}_t|s_t)) \tag{6}$$

In addition to each NLL loss, we also train the safe action distribution $\mathcal{N}^s$ on the KL-divergence loss. Since the aim is that the safe action distribution should avoid overlap with the actions that could lead to collisions, we maximise the KL-divergence between the two distributions. Therefore, the optimal parameters θ^* are obtained by the following three losses:

$$\theta^* = \arg\min_{\theta^s} \sum_t \mathcal{L}_{NLL}^s(\mathcal{N}^s, \hat{a}_t|s_t \sim \mathcal{D}^e)$$
$$+ \mathcal{L}_{NLL}^c(\mathcal{N}^c, \hat{a}_t|s_t \sim \mathcal{D}^c) - D_{KL}(\mathcal{N}^s|\mathcal{N}^c, s_t \sim \mathcal{D}^c) \tag{7}$$

The hyperparameters used for training the AMDN are given in Table I. Each neuron in the hidden layer uses the Rectified Linear Unit (ReLU) activation function, whilst each μ output uses a tanh activation function, and each σ^2 output uses a Non-negative Exponential Linear Unit (NNeLU) activation. Each loss $\mathcal{L}_{NLL}^s$, $\mathcal{L}_{NLL}^c$, D_{KL} uses learning rates η_s, η_c, η_{KL}, respectively. All networks were trained using the Adam [37] optimiser.

TABLE II: Testing of learned control policies under Natural (Nat.) and Adversarial (Adv.) Testing frameworks, with baseline comparison including Imitation Learning with Feed-Forward Network, a standard Mixture Density Network, and different versions of Adversarial Mixture Density Networks.

Testing Framework	Parameter	FFN [35]	MDN	AMDN (w/o D_{KL})	AMDN (sampling)	AMDN
Nat. Testing	min. x_{rel} [m]	23.84	0.00	0.00	7.66	11.34
	mean x_{rel} [m]	57.37	54.57	55.75	57.55	56.29
	max. v_{rel} [m/s]	8.88	14.56	14.21	11.79	10.93
	mean v_{rel} [m/s]	0.0197	0.0126	0.0138	0.0219	0.0224
	min. t_h [s]	1.74	0.00	0.00	0.90	1.13
	mean t_h [s]	1.99	1.90	1.94	2.00	1.95
	collisions	0	1	2	0	0
Adv. Testing	collisions against adversaries	800	63	51	1	0
	episodes until collision	245	684	1998	1974	-

III. RESULTS

We test the effectiveness of the proposed AMDN approach in two different evaluation frameworks. *Naturalistic testing* tests the control policy in common driving scenarios which represent the typical scenarios encountered when driving on highways. These tests are completed in 5-minute episodes with pre-defined lead vehicle trajectories in the IPG Car-Maker Simulator. The types of lead vehicle trajectories seen are similar to those seen in the expert driver dataset $\mathcal{D}^e$. For each model, we test the control policy in 120 different scenarios, which totals up to 10 hours of driving. In contrast, *adversarial testing* utilises the adversarial agents presented in [14] to create safety-critical scenarios. The aim of these agents is to act in such a way that the vehicle follower should collide into them. However, the actions of the adversarial agents are limited such that any collisions would have been preventable, and therefore represent a fault in the vehicle follower's control policy. We utilise the lead vehicle velocity and acceleration limits which were shown to be most effective for creating collisions in highway driving in [14], such that the velocity and acceleration are limited to $v_{lead} \in [12, 30]$ m/s and $a_{lead} \in [-6, 2]$ m/s^2, respectively. For each model tested, we train 5 adversarial agents for up to 2,500 episodes, and then report the number of collisions averaged over the 5 training runs. It is worth noting that, while the adversarial testing framework is the same as the one used to collect the dataset $\mathcal{D}^c$, each of the adversaries used in the testing in this Section is randomly initialised and trained specifically against one network only. If the same adversarial agents that were used to collect the dataset would be used for validation purposes, the agents could encounter similar adversarial behaviour seen during training. Instead, these adversarial agents are not the same agents as those that collected the dataset $\mathcal{D}^c$, and the adversarial testing demonstrates how easy it is for new adversaries to find control strategies that can exploit weaknesses in the target policies. For both types of testing, at the beginning of each

test episode, a coefficient of friction between 0.4 and 1.0 is randomly chosen.

We compare the AMDN to two different baselines, FFN is the Feed-Forward Network based Imitation Learning control policy presented in [35] and MDN is a standard Mixture Density Network with $M = 1$, which outputs a single Gaussian Distribution trained on dataset $\mathcal{D}^e$. During inference, all MDN and AMDN variants, unless otherwise stated, use the output μ^s as the pedal action, since during testing it was shown that this significantly improves the stability of the vehicle control policy. To compare this inference strategy to sampling from the distribution $\mathcal{N}^s$, we provide comparison to the AMDN (sampling) model which samples action from the Gaussian distribution during inference. We also compare the model to the AMDN without the KL-divergence loss, which is denoted by AMDN (w/o D_{KL}).

The summary of the results from the naturalistic testing are provided in the top half of Table II, while the results of the adversarial testing are shown in lower half of the table. Comparing the AMDN to the AMDN (sampling) model shows that using the μ^s as the action during deployment is a better inference strategy, as the AMDN policy using μ^s as the action during inference maintains a safer distance from the lead vehicle. Moreover, when sampling the pedal action from the $\mathcal{N}^s$ distribution, the variance of the sampled actions creates jerky driving trajectories with rapid accelerations and decelerations, which can present safety and passenger comfort problems when driving. However, it is still beneficial to model the two Gaussian distributions during training, as this allows the use of the KL-divergence loss on the unlabelled collision dataset. The benefit of the D_{KL} loss can be seen by comparing the performance of the AMDN to the AMDN (w/o D_{KL}) model. Not only does the AMDN (w/o D_{KL}) collide 2 times in the naturalistic driving, but it shows significantly greater susceptibility to adversarial testing as shown by its 51 collisions, demonstrating the significant benefit of using the KL-divergence loss during training. Moreover, comparing the AMDN to the standard
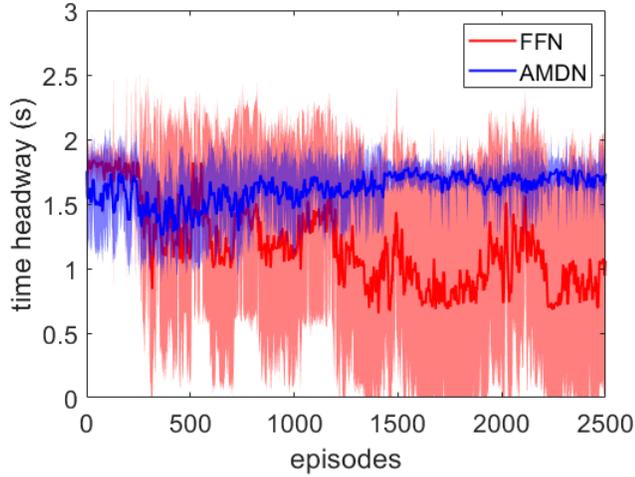
Fig. 2: Minimum episode headway during Adversarial Testing. Averaged over 5 training runs, with standard deviation shown in the shaded region.
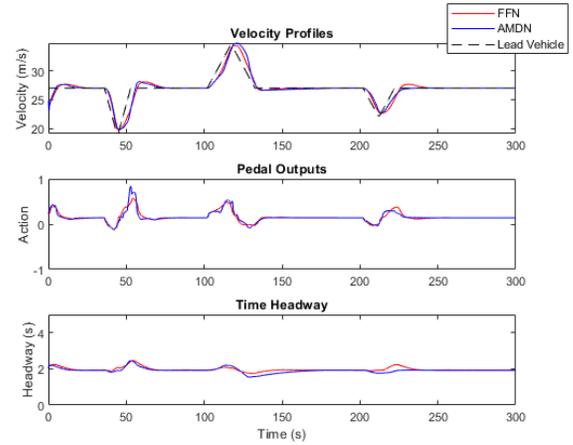

Fig. 3: Example vehicle following scenario from naturalistic driving tests. Vehicle velocities (top), host vehicle pedal actions (middle), and relative time headway (bottom).

MDN also shows the benefit of the proposed approach, as the AMDN shows significantly better driving performance in both the naturalistic and adversarial testing compared to MDN. Of the models compared here, the only two models that can drive without collisions in naturalistic driving are the FFN and AMDN. The results demonstrate that while the two models can drive safely without collisions, the FFN performs slightly better in naturalistic driving as it has slightly higher minimum headway at 1.74s compared to 1.13s and a minimum distance of 23.84m compared to 11.34m. However, on average the models show no significant difference as the FFN has a mean headway of 1.99s compared to 1.95s for AMDN, and the mean distances are 57.37m compared to 56.29m. However, comparing the FFN and AMDN during adversarial testing shows one of the main benefits of the AMDN approach. The FFN shows significant vulnerability when facing unpredictable driving behaviour different from its training distribution, as demonstrated by its average number of 800 collisions in adversarial testing, whilst the AMDN can still drive without collisions even in the presence of adversarial agents that attempt to deliberately cause collisions. The minimum headway during adversarial testing can be seen in Fig. 2, where the results are averaged over the 5 adversarial agents trained against each policy. As can be seen the minimum headway for the FFN policy reduces over the training time with increasing variance as the adversarial agents learn to exploit the FFN model. In comparison, the minimum headway for AMDN remains over 1s without reducing significantly, suggesting the adversarial agents are struggling to learn how to exploit the AMDN thus demonstrating increased robustness to unpredictable and adversarial road users.

In addition to the summaries of the different tests, we also show some example episodes in Fig. 3 - 5 comparing the learned driving policies by the FFN and AMDN models under naturalistic driving. The example scenario in Fig. 3 shows

the lead vehicle starting at a constant velocity, and then decelerating rapidly following by immediately accelerating and vice versa. We see that both the FFN and AMDN models can follow this type of trajectories easily without straying far from the target headway of 2s. The control policies' response to an emergency braking manoeuvre by the lead vehicle is shown in Fig. 4. The scenario occurs in low friction conditions with a road friction coefficient of 0.475. At $t = 224$s, the lead vehicle suddenly decelerates at 4m/s^2, forcing the host vehicle to brake in response. The vehicle then drives at 13m/s for a period, followed by accelerating at 1.5m/s^2 back to its original velocity. As can be seen in Fig. 4, both the FFN and AMDN models can adequately respond to this harsh braking in low friction conditions, and avoid any potential collision without bringing the vehicle too close to the lead vehicle. To demonstrate the worst case behaviour by the AMDN, in Fig. 5 the test scenario in which the AMDN control policy demonstrated its lowest headway in naturalistic testing is presented. This scenario includes multiple accelerations and decelerations by the lead vehicle. We can see that both vehicle followers again maintain a safe distance from the lead vehicle. However, at two points in the scenario we can see the lead vehicle decelerate to a lower velocity, and after some time accelerating back to its original velocity. As the lead vehicle starts to accelerate in this manoeuvre, we can see the AMDN decide to engage the brakes of the vehicle, followed by rapid acceleration to catch up to the lead vehicle. While this behaviour does not result in any safety-critical situation, this type of jerky manoeuvre could present comfort concerns for any passengers. However, these models were not trained explicitly to consider passenger comfort, therefore expanding this work to include considerations for passenger comfort could be included in future work.

Investigating the behaviour of the two models in the adversarial testing can provide further insight into the learned policies. An example collision episode from the adversarial testing of the FFN is shown in Fig. 6. As can be seen,
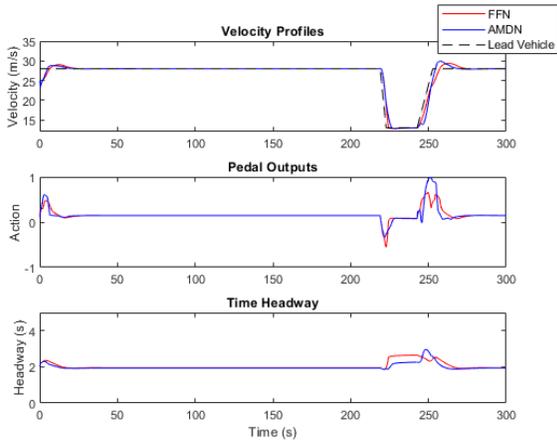
Fig. 4: Example vehicle following scenario from naturalistic driving tests. Vehicle velocities (top), host vehicle pedal actions (middle), and relative time headway (bottom).
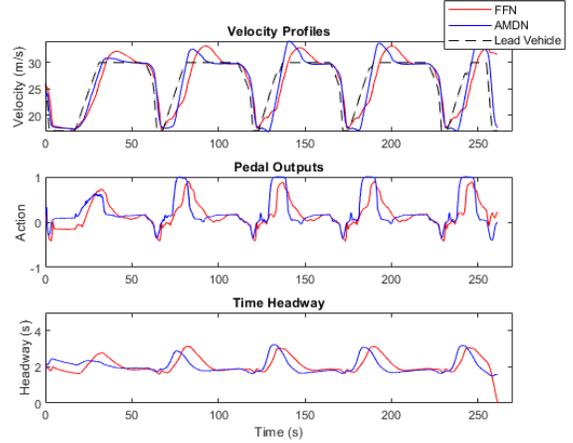

Fig. 6: Example vehicle following scenario from adversarial testing. Vehicle velocities (top), host vehicle pedal actions (middle), and relative time headway (bottom).
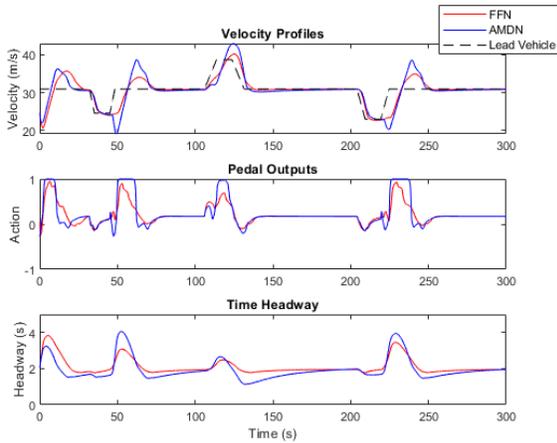

Fig. 5: Example vehicle following scenario from naturalistic driving tests. Vehicle velocities (top), host vehicle pedal actions (middle), and relative time headway (bottom).
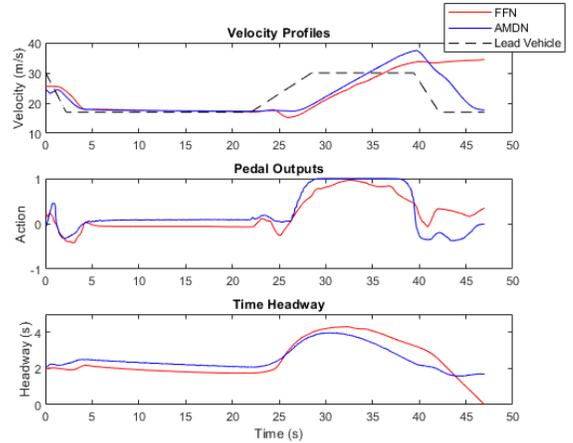

Fig. 7: Example vehicle following scenario from adversarial testing. Vehicle velocities (top), host vehicle pedal actions (middle), and relative time headway (bottom).

the adversarial lead vehicle keeps decelerating to a low velocity followed by accelerating to a high velocity, and then continues to repeat this manoeuvre. While the AMDN keeps a safe distance with a minimum headway of 1.48s, the FFN eventually fails to maintain a safe distance as it accelerates towards the lead vehicle to catch up but the lead vehicle suddenly breaks harshly and a collision occurs. Another example from the adversarial testing is seen in Fig. 7, where again the adversarial lead vehicle starts from a low velocity and then rapidly accelerates to a high velocity. As the follower vehicle is accelerating to catch the lead vehicle, the adversary then breaks suddenly. However, this scenario shows that even when the lead vehicle is already decelerating rapidly, the FFN agent continues to accelerate because the vehicles have a large relative headway and the FFN is trying to reach a headway of 2s. Meanwhile, the AMDN agent responds to the rapid deceleration by the lead vehicle by also braking, even though the headway between

the agents is above 2s. These results suggest that the FFN model, trained only on the expert demonstrated data where a 2s headway was always followed, considers the headway more important while the AMDN focuses on maintaining the same speed with the lead vehicle. This demonstrates a difference in the learned driving strategies of the two models, and also provides further insight into why the AMDN has lower minimum headways compared to FFN in naturalistic driving, despite demonstrating better capabilities at avoiding collisions. Indeed, this could be a driving strategy it learned from the collision dataset $\mathcal{D}^c$, which helps the model be robust to these types of adversarial agents and maintain a collision free driving performance.

IV. CONCLUSIONS

In this paper, a technique for learning safe control policies for autonomous driving was presented. The proposed technique is based on Mixture Density Networks, where the model learns two separate action distributions. The first

distribution is a safe action distribution learned from a dataset of expert demonstration, whilst the second is a unsafe action distribution learned from a dataset of collisions. By learning each of these distributions, the model then uses a KL loss, which penalises the safe action distribution for similarity to unsafe action distribution, using example states from the collision dataset. In this way, the model learns a more safe and robust control policy, which remains safe even in dangerous edge-cases. The proposed approach was demonstrated in a highway vehicle following use-cases, and the Adversarial Mixture Density Networks were compared to standard imitation learning with feed-forward networks and mixture density networks. The results demonstrate that the proposed approach results in a more safe control policy, which is better able to react to unpredictable and adversarial road users.

REFERENCES

[1] A. Eskandarian, C. Wu, and C. Sun, "Research advances and challenges of autonomous and connected ground vehicles," *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[2] A. Eskandarian, *Handbook of intelligent vehicles*. Springer, 2012, vol. 2.

[3] S. Thrun, "Toward robotic cars," *Communications of the ACM*, vol. 53, no. 4, pp. 99–106, 2010.

[4] S. Kuutti, R. Bowden, Y. Jin, P. Barber, and S. Fallah, "A survey of deep learning applications to autonomous vehicle control," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[5] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Computing Surveys (CSUR)*, vol. 50, no. 2, pp. 1–35, 2017.

[6] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," in *Advances in Neural Information Processing Systems (NIPS)*, 1989, pp. 305–313.

[7] M. Bansal, A. Krizhevsky, and A. Ogale, "Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst," *arXiv preprint arXiv:1812.03079*, 2018.

[8] D. Wang, C. Devin, Q.-Z. Cai, F. Yu, and T. Darrell, "Deep object centric policies for autonomous driving," *arXiv preprint arXiv:1811.05432*, 2018.

[9] S. Hecker, D. Dai, and L. Van Gool, "End-to-end learning of driving models with surround-view cameras and route planners," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 435–453.

[10] J. A. Bagnell, "An invitation to imitation," CARNEGIE-MELLON UNIV PITTSBURGH PA ROBOTICS INST, Tech. Rep., 2015.

[11] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters, "An algorithmic perspective on imitation learning," *arXiv preprint arXiv:1811.06711*, 2018.

[12] F. Codevilla, A. M. Lopez, V. Koltun, and A. Dosovitskiy, "On offline evaluation of vision-based driving models," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 236–251.

[13] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *2011 International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011, pp. 627–635.

[14] S. Kuutti, S. Fallah, and R. Bowden, "Training adversarial agents to exploit weaknesses in deep control policies," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 108–114.

[15] P. de Haan, D. Jayaraman, and S. Levine, "Causal confusion in imitation learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019, pp. 11 693–11 704.

[16] J. Ho and S. Ermon, "Generative adversarial imitation learning," in *Advances in Neural Information Processing Systems (NIPS)*, 2016, pp. 4565–4573.

[17] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems (NIPS)*, 2014, pp. 2672–2680.

[18] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep reinforcement learning framework for autonomous driving," *Electronic Imaging*, vol. 2017, no. 19, pp. 70–76, 2017.

[19] L. Puccetti, C. Rathgeber, and S. Hohmann, "Actor-critic reinforcement learning for linear longitudinal output control of a road vehicle," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 2907–2913.

[20] H. Chae, C. M. Kang, B. Kim, J. Kim, C. C. Chung, and J. W. Choi, "Autonomous braking system via deep reinforcement learning," in *2017 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2017, pp. 1–6.

[21] S. Kuutti, R. Bowden, H. Joshi, R. de Temple, and S. Fallah, "End-to-end reinforcement learning for autonomous longitudinal control using advantage actor critic with temporal context," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 2456–2462.

[22] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, "Deep reinforcement learning that matters," *arXiv preprint arXiv:1709.06560*, 2017.

[23] R. A. Fisher, "The statistical utilization of multiple measurements," *Annals of eugenics*, vol. 8, no. 4, pp. 376–386, 1938.

[24] K. Fukunaga, *Introduction to statistical pattern recognition*. Elsevier, 2013.

[25] A. M. Martinez and A. C. Kak, "Pca versus lda," *IEEE transactions on pattern analysis and machine intelligence*, vol. 23, no. 2, pp. 228–233, 2001.

[26] D. A. Pomerleau, "Efficient training of artificial neural networks for autonomous navigation," *Neural computation*, vol. 3, no. 1, pp. 88–97, 1991.

[27] C. M. Bishop, "Mixture density networks," 1994.

[28] G. J. McLachlan and K. E. Basford, *Mixture models: Inference and applications to clustering*. M. Dekker New York, 1988, vol. 38.

[29] X. Wang, S. Takaki, and J. Yamagishi, "An autoregressive recurrent mixture density network for parametric speech synthesis," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 4895–4899.

[30] O. Makansi, E. Ilg, O. Cicek, and T. Brox, "Overcoming limitations of mixture density networks: A sampling and fitting framework for multimodal future prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7144–7153.

[31] Y. Hu, W. Zhan, and M. Tomizuka, "Probabilistic prediction of vehicle semantic intention and motion," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 307–313.

[32] S. Choi, K. Lee, S. Lim, and S. Oh, "Uncertainty-aware learning from demonstration using mixture density networks with sampling-free variance modeling," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 6915–6922.

[33] M. Yan, A. Li, M. Kalakrishnan, and P. Pastor, "Learning probabilistic multi-modal actor models for vision-based robotic grasping," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 4804–4810.

[34] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[35] S. Kuutti, R. Bowden, H. Joshi, R. de Temple, and S. Fallah, "Safe deep neural network-driven autonomous vehicles using software safety cages," in *2019 20th International Conference on Intelligent Data Engineering and Automated Learning (IDEAL)*. Springer, 2019, pp. 150–160.

[36] IPG Automotive GmbH, "Carmaker: Virtual testing of automobiles and light-duty vehicles," 2017. [Online]. Available: https://ipg-automotive.com/products-services/simulation-software/carmaker/

[37] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.