Safe Reinforcement Learning With Model Uncertainty Estimates

Safe Reinforcement Learning with Model Uncertainty Estimates

Björn Lütjens, Michael Everett, Jonathan P. How

***Abstract*— Many current autonomous systems are being designed with a strong reliance on black box predictions from deep neural networks (DNNs). However, DNNs tend to be overconfident in predictions on unseen data and can give unpredictable results for far-from-distribution test data. The importance of predictions that are robust to this distributional shift is evident for safety-critical applications, such as collision avoidance around pedestrians. Measures of model uncertainty can be used to identify unseen data, but the state-of-the-art extraction methods such as Bayesian neural networks are mostly intractable to compute. This paper uses MC-Dropout and Bootstrapping to give computationally tractable and parallelizable uncertainty estimates. The methods are embedded in a Safe Reinforcement Learning framework to form uncertainty-aware navigation around pedestrians. The result is a collision avoidance policy that *knows what it does not know* and cautiously avoids pedestrians that exhibit unseen behavior. The policy is demonstrated in simulation to be more robust to novel observations and take safer actions than an uncertainty-unaware baseline.**

I. Introduction

Reinforcement learning (RL) is used to produce state-of-the-art results in manipulation, motion planning and behavior prediction. However, the underlying neural networks often lack the capability to produce qualitative predictive uncertainty estimates and tend to be overconfident on out-of-distribution test data [1]–[3]. In safety-critical tasks, such as collision avoidance of cars or pedestrians, incorrect but confident predictions of unseen data can lead to fatal failure [4]. We investigate methods for Safe RL that are robust to unseen observations and *know what they do not know* to be able to raise an alarm in unpredictable test cases; ultimately leading to safer actions.

A particularly challenging safety-critical task is avoiding pedestrians in a campus environment with an autonomous shuttle bus or rover [5], [6]. Humans achieve mostly collision-free navigation by understanding the hidden intentions of other pedestrians and vehicles and interacting with them [7], [8]. Furthermore, most of the time this interaction is accomplished without verbal communication. Our prior work uses RL to capture the hidden intentions and achieve collaborative navigation around pedestrians [9]–[11]. However, RL approaches always face the problem of generalizability from simulation to the real world and cannot guarantee performance on far-from-training test data. An example policy that has only been trained on collaborative pedestrians could fail to generalize to uncollaborative pedestrians in the real world, as seen in Section I. The trained policy would output a best guess policy that might assume collaborative behavior and, without labeling the novel observation, fail ungracefully. To avoid such failure cases, this paper develops a Safe RL framework for dynamic collision avoidance that expresses novel observations in the form of model uncertainty. The framework further reasons about the uncertainty and cautiously avoids regions of high uncertainty, as displayed in Fig. 6.

Aerospace Controls Laboratory, Massachusetts Institute of Technology, 77 Mass. Ave., Cambridge, MA, USA {lutjens, mfe, jhow}@mit.edu

Fig. 1: An autonomous vehicle observes a novel dynamic obstacle that has never appeared during training, for example, an uncollaborative pedestrian on a personal vehicle. The proposed Reinforcement Learning framework detects the novelty and takes an action that cautiously avoids the pedestrian.

Much of the existing Safe RL research has focused on using external novelty detectors or internal modifications to identify environment or model uncertainty [12]. Note that our work targets *model uncertainty* estimates because they potentially reveal sections of the test data where training data was sparse and a model could fail to generalize [13]. Work in risk-sensitive RL (RSRL) often focuses on *environment uncertainty* to detect and avoid high-risk events that are known from training to have low probability but high cost [14]–[18]. Other work in RSRL targets model uncertainty in MDPs, but does not readily apply to neural networks [15], [19]. Our work is mainly orthogonal to risk-sensitive RL approaches and could be combined into an RL policy that is robust to unseen data and sensitive to high-risk events.

Extracting model uncertainty from discriminatively trained neural networks is complex, as the model outcome for a given observation is deterministic. Mostly, Bayesian neural networks are used to extract model uncertainty but require a significant restructuring of the network architecture [20]. Additionally, even approximate forms, such as Markov Chain Monte Carlo [20] or variational methods [21]–[23], come with extensive computational cost and have a sample-dependent accuracy [2], [20], [24]. Our work uses Monte Carlo Dropout

(MC-Dropout) [25] and bootstrapping [26] to give parallelizable and computationally feasible uncertainty estimates of the neural network without significantly restructuring the network architecture [27], [28].

The main contributions of this work are i) an algorithm that identifies novel pedestrian observations and ii) avoids them more cautiously and safer than an uncertainty-unaware baseline, iii) an extension of an existing uncertainty-aware reinforcement learning framework [29] to more complex dynamic environments with exploration aiding methods, and iv) a demonstration in a simulation environment.

II. Related Work

This section investigates related work in Safe Reinforcement Learning to develop a dynamic collision avoidance policy that is robust to out-of-data observations.

A. External verification and novelty detection

Many related works use off-policy evaluation or external novelty detection to verify the learned RL policy [12], [30], [31]. Reachability analysis could verify the policy by providing regional safety bounds, but the bounds would be too conservative in a collaborative pedestrian environment [32]–[35]. Novelty detection approaches place a threshold on the detector's novelty output and switch to a safety controller if the threshold is exceeded [30]. However, switching to safety controllers is often abrupt and can generate uncomfortable, and unpredictable driving behavior. In our framework, the vehicle stays away from uncertain regions, as seen in Fig. 3, to predictively avoid interventions by an underlying safety controller.

B. Environment and model uncertainty

This paper focuses on detecting novel observations via model uncertainty, also known as parametric or epistemic uncertainty [36]. The orthogonal concept of environment uncertainty captures the uncertainty due to the imperfect nature of partial observations [13]. For example, an observation of a pedestrian trajectory will, even with infinite training in the real-world, not fully capture the decision-making process of pedestrians and thus be occasionally ambiguous; will she turn left or right? The RL framework accounts for the unobservable decision ambiguity by learning a mean outcome [13]. Model uncertainty, in comparison, captures how well a model fits all possible observations from the environment. It could be explained away with infinite observations and is typically high in applications with limited training data, or with test data that is far from the training data [13]. Thus, the model uncertainty captures cases in which a model fails to generalize to novel test data and hints when one should not trust the network predictions [13].

C. Measures of model uncertainty

A new research topic adapts neural networks to express their model uncertainty [21], [25], [26]. Bootstrapping has been explored to generate approximate uncertainty measures to guide exploration [26]. By training an ensemble of networks on partially overlapping dataset samples they agree in areas of common data and disagree, and have a large sample variance, in regions of uncommon data [2], [26]. Dropout can be interpreted similarly, if it is activated during test-time, and has been shown to approximate Bayesian inference in Gaussian processes [25], [27]. An alternative approach uses a Hypernet, a network that learns the weights of another network to directly give parameter uncertainty values, but was shown to be computationally very expensive [37]. An innovative, but controversial, approach retrieves Bayesian uncertainty estimates via batch normalization [38]. This work uses MC-Dropout and bootstrapping to give computationally tractable uncertainty estimates.

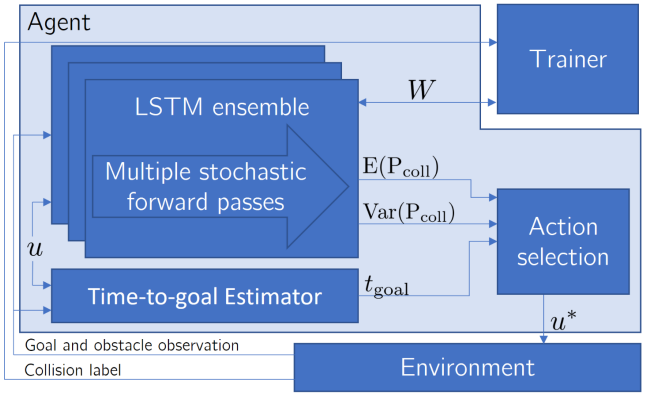


Fig. 2: System architecture. An agent observes the environment and selects minimal cost motion primitives u^* to reach a goal while avoiding collisions. On each time step, an ensemble of LSTM networks is sampled multiple times with different dropout masks to acquire a sample mean and variance collision probability for each motion primitive u.

D. Applications of model uncertainty in RL

Measures of model uncertainty have been used in RL very recently to speed up training by guiding the exploration into regions of high uncertainty [26], [39], [40]. Kahn et al. used uncertainty estimates in model-based RL for static obstacle collision avoidance [29]. Instead of a model-based RL approach, one could argue to use model-free RL and draw the uncertainty of an optimal policy output $\pi^* = \mathrm{argmax}_\pi(Q)$. However, the uncertainty estimate would contain a mix from the uncertainties of multiple objectives and would not focus on the uncertain region of collision. Our work extends the model-based framework by [29] to the highly complex domain of pedestrian collision avoidance. [29] is further extended by using the uncertainty estimates for guided exploration to escape locally optimal policies, analyzing the regional increase of uncertainty in novel dynamic scenarios, using LSTMs, and acting goal-guided.

III. Approach

This work proposes an algorithm that uses uncertainty information to cautiously avoid dynamic obstacles in novel scenarios. As displayed in the system architecture in Fig. 2, an agent observes a simulated obstacle's position and velocity, and the goal. A set of Long-Short-Term-Memory (LSTM) [41]

networks predicts collision probabilities for a set of motion primitives u. MC-Dropout and bootstrapping are used to acquire a distribution over the predictions. From the predictions, a sample mean $\mathrm{E}(\mathrm{P}_{\text{coll}})$ and variance $\mathrm{Var}(\mathrm{P}_{\text{coll}})$ is drawn for each motion primitive. In parallel, a simple model estimates the time to goal t_{goal} at the end of each evaluated motion primitive. In the next stage, the minimal cost motion primitive u^* is selected and executed for one step in the environment. The environment returns the next observation and at the end of an episode a collision label. After a set of episodes, the network weights W are adapted and the training process continues. Each section of the algorithm is explained in detail below.

A. Collision Prediction Network

A set of LSTM networks (ensemble) predicts the collision probabilities of motion primitives. Each forward pass i of a network returns the collision probability of an evaluated motion primitive:

$$\mathrm{P}^i_{\text{coll}} = \mathrm{P}^i\left(\mathbb{1}_{\text{coll}} = 1 | o_{t-l:t-1}, o_t, u_{t-l:t-1}, u_{t:t+h}\right)$$

where $\mathbb{1}_{\text{coll}}$ is a collision label; $o_{t-l:t-1}$ is the history of observations in the last l time steps; o_t is the current observation; $u_{t-l:t-1}$ is a concatenation of past actions; and $u_{t:t+h}$ is the evaluated motion primitive of length h. The RL agent operates in a partially observable environment where it can only observe the pedestrian's position, velocity, and radius. The observation further contains the relative goal position of the RL agent. The motion primitive $u_{t:t+h}$ is element of a precomputed set of motion primitives U. In this work, U contains 11 discrete motion primitives of length $h = 1$ which are described by a heading angle $\alpha \in [-\frac{\pi}{6}, \frac{\pi}{6}]$. Regardless of the length, the optimal motion primitive is taken for one time step until the network is queried again.

LSTM networks are chosen for the dynamic obstacle avoidance, because they are the state-of-the-art model in predicting pedestrian paths by understanding the hidden temporal intentions of pedestrians best [42], [43]. Based on this success, the proposed work first applies LSTMs to pedestrian avoidance in an RL setting. For safe avoidance, LSTM predictions need to be accurate from the first time step a pedestrian is observed in the robot's field of view. To handle the variable length observation input, masking [44] is used during training and test to deactivate LSTM cells that exceed the length of the observation history.

B. Uncertainty Estimates with MC-Dropout and Bootstrapping

MC-Dropout [25] and bootstrapping [2], [26] are used to compute stochastic estimates of the model uncertainty $\mathrm{Var}(\mathrm{P}_{\text{coll}})$. For bootstrapping, multiple networks are trained and stored in an ensemble. Each network is randomly initialized and trained on sample datasets that have been drawn with replacement from a bigger experience dataset [26]. By being trained on different but overlapping sections of the observation space, the network predictions differ for uncommon observations and are similar for common observations. As each network can be trained and tested in parallel, bootstrapping does not come with significant computational cost and can be run on a real robot.

Dropout [27] is traditionally used for regularizing networks. It randomly deactivates network units in each forward pass by multiplying the unit weights with a dropout mask. The dropout mask is a set of Bernoulli random variables of value $[0, 1]$, each with a keeping probability p. Traditionally, dropout is deactivated during test and each unit is multiplied with p. However, [25] has shown that an activation of dropout during test, named MC-Dropout, gives model uncertainty estimates by approximating Bayesian inference in deep Gaussian processes. To retrieve the model uncertainty with dropout, our work executes multiple forward passes per network in the bootstrapped ensemble with different dropout masks ($p = 0.7$) and acquires a distribution over predictions. For n_d dropout samples in n_b networks, a total of $N = n_d n_b$ forward passes are sampled. Although dropout has been seen to be overconfident on novel observations [26], Table I shows that the combination of bootstrapping and dropout reliably detects novel scenarios.

From the parallelizable collision predictions from each network and each dropout mask, the sample mean and variance is drawn.

C. Selecting actions

A Model Predictive Controller (MPC) selects the safest motion primitive with the minimal joint cost:

$$u^\star_{t:t+h} = \underset{u \in U}{\operatorname{argmin}} \left(\lambda_v \mathrm{Var}_N(\mathrm{P}^i_{\text{coll}}) + \ \lambda_c \mathrm{E}_N(\mathrm{P}^i_{\text{coll}}) + \lambda_g t_{\text{goal}}\right)$$

The chosen MPC that considers the second order moment of probability [29], [45], [46] is able to select actions that are more certainly safe. The first and second order moment ($\mathrm{E}(\cdot)$ and $\mathrm{V}ar(\cdot)$) are computed over the N forward passes per motion primitive. The MPC estimates the time-to-goal t_{goal} from the end of each motion primitive by measuring the straight line distance. Each cost term is weighted by its own factor λ. Note that the soft constraint on collision avoidance requires λ_g and λ_c to be chosen such that the predicted collision cost $\lambda_c \mathrm{E}_N(\mathrm{P}^i_{\text{coll}})(\leq \lambda_c)$ is greater than the goal cost $\lambda_g t_{\text{goal}}$. In comparison to [29], this work does not multiply the variance term with the selected velocity. The reason being is that simply stopping or reducing one's velocity is not always safe, for example on a highway scenario or in the presence of adversarial agents. The proposed work instead focuses on identifying and avoiding uncertain observations regionally in the ground plane.

D. Adaptive variance

Note that during training an overly uncertainty-averse model would discourage exploration and rarely find the optimal policy. Additionally, the averaging over multiple forward passes during prediction reduces the ensemble's diversity, which additionally hinders explorative actions. The proposed approach increases the penalty on highly uncertain actions λ_v over time to overcome this effect. Thus, the policy efficiently explores in directions of high model uncertainty

during early training phases; λ_v is brought to convergence to act uncertainty-averse during execution. This work linearly increases λ_v in $[-50000, 200]$ and has $\lambda_g = 2$, and $\lambda_c = 25$.

E. Collecting the dataset

The selected action is executed in the learning environment. At the end of each episode t_{end}, the environment returns a collision label $\mathbb{1}_{\text{coll}}$. The collision label is one if a collision occured during the episode and zero otherwise. The history of observations $o_{t_{start}:t_{\text{end}}}$ and actions $u_{t_{start}:t_{\text{end}}}$ from start to end of an episode is associated with the collision label and stored in an experience dataset. After running several episodes, random subsets from the full experience set are drawn to train the ensemble of networks for the next observe-act-train cycle. The policy roll-out cycle is necessary to learn how dynamic obstacles will react to the agent's learned policy. A supervised learning approach, as taken in [30] for static obstacle avoidance, would not learn the reactions of environment agents on the trained policy.

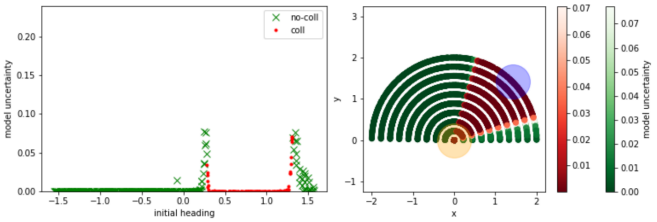
IV. Results

We show that our algorithm uses uncertainty information to regionally detect novel obstacle observations and causes fewer collisions than an uncertainty-unaware baseline. First, a simple 1D case illustrates how the model regionally identifies novel obstacle observations. In a scaled up environment with novel multi-dimensional observations, the proposed model continues to exhibit regionally increased uncertainty values. The model is compared with an uncertainty-unaware baseline in a variety of novel scenarios; the proposed model performs more robust to novel data and causes fewer collisions.

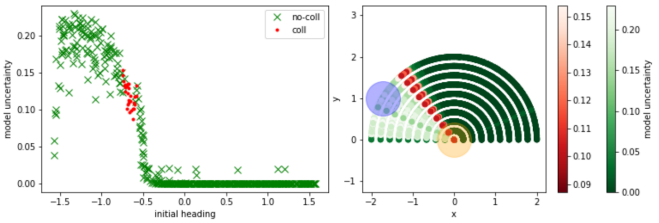
A. Regional novelty detection in 1D

First, we show that model uncertainty estimates are able to detect novel one-dimensional observations regionally, as seen in Fig. 3. For the 1D test-case, a two-layer fully-connected network with MC-Dropout and Bootstrapping is trained to predict collision labels. To generate the dataset, an agent randomly chose heading actions, independent of the obstacle observations, and the environment reported the collision label. The network input is the agent heading angle and obstacle heading. Importantly, the training set only contains obstacles that are on the right-hand side of the agent (top plot:$x > 0$).

After training, the network accurately predicts collision and no-collision labels with low uncertainty for obstacle observations from the training distribution, as seen in Fig. 3a. For out-of-training obstacle observations on the agent's left (bottom plot: $x < 0$), the neural network fails to generalize and predicts collision (red) as well as non-collision (green) labels for actions (straight lines) that would collide with the obstacle (blue). However, the agent identifies regions of high model uncertainty (left: y-axis, right: light colors) for actions in the direction of the unseen obstacle. The high uncertainty values suggest that the network predictions are false-positives and should not to be trusted. Based on the left-right difference in uncertainty estimates, the MPC would prefer a conservative action that is certainly safe (bottom-right: dark green lines)

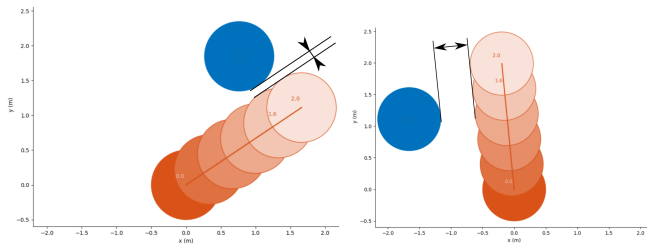


(a) Known obstacle: low uncertainty



(b) Novel obstacle: high uncertainty

Fig. 3: Regional novelty detection in 1D. A simple network predicts collision (red) and no-collision (green) labels, given the agent's (orange) heading (left plot: x-axis) and a one-dimensional observation of an obstacle (blue) heading. The network accurately predicts labels with low uncertainty, when tested on the training dataset (a) . When tested on a novel observation set (b), the networks fails to predict accurate decision labels, but identifies them with a high regional uncertainty (bottom-left: green points with high values, bottom-right: light green lines). Rather than believing in the false-positive collision predictions, Figure 4 depicts how an agent would take a certainly safe action (dark green) to cautiously avoid the novel obstacle.



(a) Known obstacle, confident (b) Novel obstacle, cautious

Fig. 4: Cautious avoidance after regional novelty detection. An agent (orange) in Fig. 4a uses the uncertainty estimates from Fig. 3a to avoid a known obstacle (blue) confidently close. In Fig. 4b, an agent recognizes a novel obstacle appearance, as seen in Fig. 3b, and cautiously avoids the obstacle.

over a false-positive action that is predicted to be safe but uncertain (bottom-right: light green lines). Figure 4 illustrates how the MPC chooses a conservative action to avoid a novel obstacle and confident actions to avoid known obstacles.

B. Novelty detection in multi-dimensional observations

The following experiments show that our model continues to regionally identify uncertainty in multi-dimensional observations and choose safer actions.

1) Experiment setup: A one-layer 16-unit LSTM model has been trained in a *gym* [47] based simulation environment with one agent and one dynamic obstacle. The dynamic obstacle in the environment is capable of following a collaborative RVO [48], GA3C-CADRL [11], or non-cooperative or static policy. For the analyzed scenarios, the agent was trained

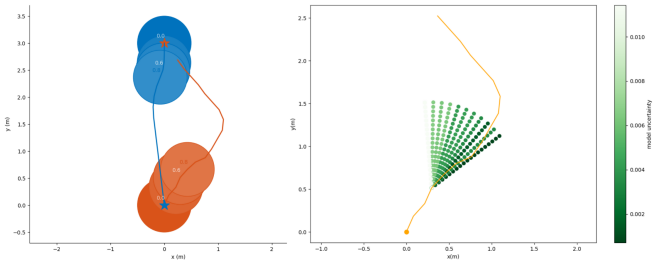
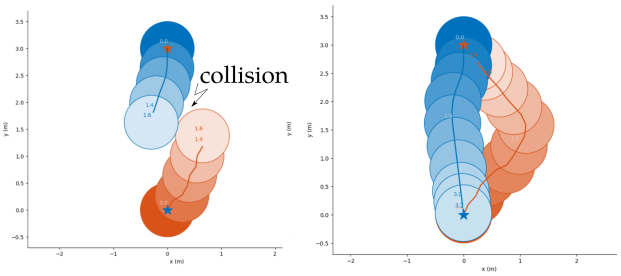


Fig. 5: Regional identification of uncertainty. An uncertainty-aware agent (orange) avoids a dynamic obstacle (blue) that is observed with noise. At one time step, collision predictions for actions in the direction of the obstacle (light green lines) are assigned a higher uncertainty than for actions in free space (dark green lines). The agent selects an action with low uncertainty to cautiously avoid the obstacle.



(a) uncertainty-unaware (b) uncertainty-aware

Fig. 6: Cautious avoidance in novel scenarios. An agent (orange) is trained to avoid dynamic RVO agents (blue) that are observed without noise. On test, Gaussian noise is added to the observation and an uncertainty-unaware model in Fig. 6a fails to generalize and causes a collision. The proposed uncertainty-aware agent in Fig. 6b acts more cautiously on novel observations and avoids the obstacle successfully.

with obstacles that follow an RVO policy and are observed as described in Section III. The training process took 20 minutes on a low-compute amazon AWS c5.large Intel Xeon Platinum 8124M with 2vCPUs and 4GiB memory. Each of the used five networks in the ensemble is sampled twenty times by stochastic MC-Dropout forward passes. Drawing in total one hundred samples per step takes in average 32ms. The train and execution time could be further decreased by parallelizing the computation on GPUs.

In the test setup, observations of obstacles are manipulated to create scenarios with novel observations that could break the trained model. In one scenario, sensor noise is simulated by adding Gaussian noise $\sim N(\mu = 0, \sigma = .5)$ on the observation of position in m and velocity in $\frac{m}{s}$. In another scenario, observations are randomly dropped with a probability of 20%. In a third and fourth scenario that simulate sensor failure, the obstacle position and velocity is masked, respectively. None of the manipulations were applied at training time.

2) Regional novelty detection: Figure 5 shows that the proposed model continues to regionally identify novel obstacle observations in a higher dimensional observation space. In the displayed experiment, an uncertainty-aware agent (orange) observes a dynamic obstacle (blue) with newly added noise and evaluates actions to avoid it. The collision predictions for actions in the direction of the obstacle (light green lines) have higher uncertainty than for actions into free-space (dark green lines). The difference in the predictive uncertainties from left to right, although being stochastic and not perfectly smooth, is used by the MPC to steer the agent away from the noisy obstacle and cautiously avoid it without a collision (orange/yellow line). Figure 6b shows the full trajectory of the uncertainty-aware agent and illustrates how an uncertainty-unaware agent in Fig. 6a with same speed and radius fails to generalize to the novel noise and collides with the obstacle after five time steps.

3) Novel scenario identification with uncertainty: Table I shows that overall model uncertainty is high in every of the tested novel scenarios, including the illustrated case of added noise. The measured uncertainty is the sum of variance of the collision predictions for each action at one time step. The

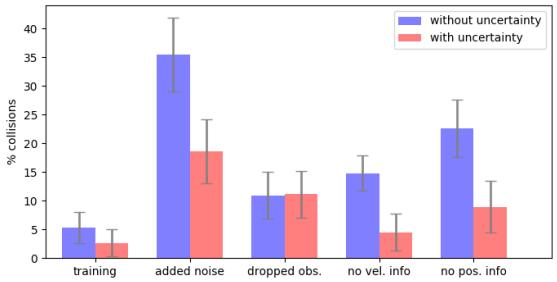


Fig. 7: Fewer collisions in novel cases. The proposed uncertainty-aware model (red) causes fewer collisions than the uncertainty-unaware baseline (blue) in novel cases. Through the regional increase of uncertainty in the obstacle's direction, the model prefers actions that more cautiously avoids the obstacle than the baseline.

uncertainty values have been averaged over 20 sessions with random initialization, 50 episodes and all time steps until the end of each episode. As seen in Table I the uncertainty in a test set of the training distribution is relatively low. All other scenarios cause higher uncertainty values and the relative magnitude of the uncertainty values can be interpreted as how novel the set of observations is for the model, in comparison to the training case.

4) Fewer collisions in novel scenarios: The proposed model uses the uncertainty information to act more cautiously and be more robust to novel scenarios. Figure 7 shows that this behavior causes fewer collisions during the novel scenarios than an uncertainty-unaware baseline. The proposed model (red) and the baseline (blue) perform similarly well on samples from the training distribution. In the test scenarios of added noise, masked position and masked velocity information, the proposed model causes fewer collisions and is more robust to the novel class of observations. In the case of dropped observations, both models perform similarly well, in terms of collisions, but the uncertainty-unaware model was seen to take longer to reach the goal. The baseline model has been trained with the same hyperparameters and environment except that the variance penalty λ_v is set to zero.

	Training	Added noise	Dropped observations	Masked vel. info.	Masked pos. info.
$E(\mathrm{Var}(\mathrm{P_{coll}}))$	0.363	0.820	1.93	1.37	2.41
$\sigma(\mathrm{Var}(\mathrm{P_{coll}}))$	0.0330	0.0915	0.134	0.0693	0.0643

TABLE I: Increased uncertainty in novel scenarios. In each of four novel test scenarios, the uncertainty of collision predictions $Var(\mathrm{P_{coll}})$ is higher than on samples from the seen training distribution.

5) Generalization to other novel scenarios: In all demonstrated cases one could have found a model that generalizes to noise, masked position observations, etc. However, one cannot design a simulation that captures all novel scenarios that could occur in real life. A significantly novel event should be recognized with a high model uncertainty. In the pedestrian avoidance task, novel observations might be uncommon pedestrian behavior, e.g. an uncollaborative pedestrian on a personal vehicle. But really all forms of observations that are novel to the deployed model should be identified and reacted upon by driving more cautiously. The shown results suggest that model uncertainty is able to identify such observations and that the MPC selects actions with extra buffer space to avoid these pedestrians cautiously.

C. Using uncertainty to escape local minima

This work increases the variance penalty λ_v to avoid getting stuck in local minima of the MPC optimization during the training process. Figure 8 shows that the proposed algorithm with increasing λ_v can escape a local minimum by encouraging explorative actions in the early stages of training. For the experiment, an agent (orange) was trained to reach a goal (star) that is blocked by a static obstacle (blue) by continuously selecting an action (left plot). In an easy avoidance case, the obstacle is placed further away from the agent's start position (in dark orange); in a challenging case closer to the agent. A close obstacle is challenging, as the agent is initially headed into the obstacle direction and needs to explore avoiding actions. The collision estimates of the randomly initialized networks are uninformative in early training stages and the goal cost drives the agent into the obstacle. A negative variance penalty λ_v in early stages forces the agent to explore actions away from the goal and avoid getting stuck in a local minimum.

Figure 8 displays that, in the challenging training case, the agent with a constant λ_v fails to explore and the algorithm gets stuck in a bad local minimum (bottom-right plot: blue), where 80% of the runs end in a collision. The policy with an increasing λ_v, and the same hyperparameters (bottom-right plot: red), is more explorative in early stages and converges to a lower minimum in an average of five sessions. In the easy test case, both algorithms perform similarly well and converge to a policy with near-zero collisions (top-right plot).

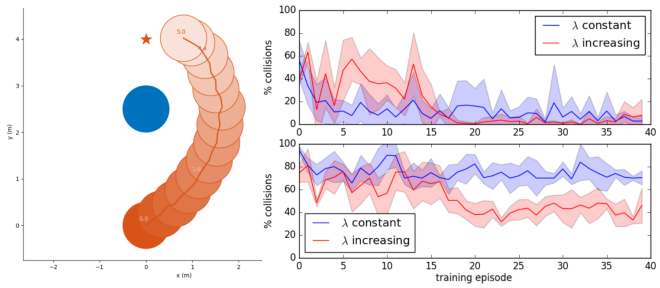


Fig. 8: Escaping local minima. The training process of two policies with a constant penalty on uncertain actions λ_v(blue) and with an increasing λ_v(red) are compared. In an easy avoidance case (right-top), both policies find a good policy that leads to near-zero collisions (y-axis). In a more challenging avoidance case (right-bottom), the proposed increasing λ_v policy, that explores in early stages, finds a better minimum than with a constant λ_v.

V. Discussion and Future Work

A. Accurately calibrated model uncertainty estimates

In another novel scenario, an agent was trained to avoid collaborative RVO agents and tested on uncollaborative agents. The uncertainty values did not significantly increase, which can be explained by two reasons. First, uncollaborative agents could not be seen as novel for the model; possibly, because RVO agents, further away from the agent also act in a straight line. The fact that humans think that uncollaborative agents might be novel for a model that has only been trained on collaborative agents, does not change the fact that the model might be generalizable enough to not see it as novel. Another explanation is the observed overconfidence of dropout as an uncertainty estimate. Future work will find unrevealed estimates of model uncertainty for neural networks that provide stronger guarantees on the true model uncertainty.

VI. Conclusion

This work has developed a Safe RL framework with model uncertainty estimates to cautiously avoid dynamic obstacles in novel scenarios. An ensemble of LSTM networks was trained with dropout and bootstrapping to estimate collision probabilities and gain predictive uncertainty estimates. The magnitude of the uncertainty estimates was shown to reveal novelties in a variety of scenarios, indicating that the model *knows what it does not know*. The regional uncertainty increase in the direction of novel obstacle observations is used by an MPC to act more cautious in novel scenarios. The cautious behavior made the uncertainty-aware framework more robust to novelties and safer than an uncertainty-unaware baseline. This work is another step towards opening up the vast capabilities of deep neural networks for the application in safety-critical tasks.

Acknowledgment

This work is supported by Ford Motor Company. The authors want to thank Golnaz Habibi for insightful discussions.

REFERENCES

[1] D. Amodei, C. Olah, J. Steinhardt, P. F. Christiano, J. Schulman, and D. Mané, "Concrete problems in AI safety," *CoRR*, vol. abs/1606.06565, 2016.

[2] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc., 2017, pp. 6402–6413.

[3] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," in *Proceedings of International Conference on Learning Representations*, 2017.

[4] The Tesla Team, "A tragic loss," June 2016. [Online]. Available: https://www.tesla.com/en_GB/blog/tragic-loss

[5] J. Miller and J. P. How, "Predictive positioning and quality of service ridesharing for campus mobility on demand systems," *2017 IEEE International Conference on Robotics and Automation*, vol. abs/1609.08116, 2016.

[6] Navya, "Navya student campus mobility," June 2018. [Online]. Available: https://navya.tech/en/autonom-en/applications/campus/

[7] Y. Zheng, T. Chase, L. Elefteriadou, B. Schroeder, and V. P. Sisiopiku, "Modeling vehicle pedestrian interactions outside of crosswalks," *Simulation Modelling Practice and Theory*, vol. 59, pp. 89 – 101, 2015.

[8] D. Helbing and P. Molnár, "Social force model for pedestrian dynamics," *Phys. Rev. E*, vol. 51, pp. 4282–4286, May 1995.

[9] Y. F. Chen, M. Liu, M. Everett, and J. P. How, "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 285–292, 2017.

[10] Y. F. Chen, M. Everett, M. Liu, and J. P. How, "Socially aware motion planning with deep reinforcement learning," *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1343–1350, 2017.

[11] M. Everett, Y. F. Chen, and J. P. How, "Motion planning among dynamic, decision-making agents with deep reinforcement learning," *CoRR*, vol. abs/1805.01956, 2018.

[12] J. García and F. Fernández, "A comprehensive survey on safe reinforcement learning," *J. Mach. Learn. Res.*, vol. 16, no. 1, pp. 1437–1480, Jan. 2015.

[13] Y. Gal, "Uncertainty in deep learning," Ph.D. dissertation, University of Cambridge, 2016.

[14] P. Geibel, "Risk-sensitive approaches for reinforcement learning," Ph.D. dissertation, University of Osnabrück, 2006.

[15] O. Mihatsch and R. Neuneier, "Risk-sensitive reinforcement learning," *Machine Learning*, vol. 49, no. 2, pp. 267–290, Nov 2002.

[16] Y. Shen, M. J. Tobia, T. Sommer, and K. Obermayer, "Risk-sensitive reinforcement learning," *Neural Comput.*, vol. 26, no. 7, pp. 1298–1328, Jul. 2014.

[17] A. Tamar, "Risk-sensitive and efficient reinforcement learning algorithms," Ph.D. dissertation, Technion - Israel Institute of Technology, Faculty of Electrical Engineering, 2015.

[18] E. Even-Dar, M. Kearns, and J. Wortman, "Risk-sensitive online learning," in *Algorithmic Learning Theory*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 199–213.

[19] Y. Chow, A. Tamar, S. Mannor, and M. Pavone, "Risk-sensitive and robust decision-making: A cvar optimization approach," in *Advances in Neural Information Processing Systems 28 (NIPS)*, 2015.

[20] R. M. Neal, *Bayesian Learning for Neural Networks*. Berlin, Heidelberg: Springer-Verlag, 1996.

[21] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural networks," in *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ser. ICML'15. JMLR, 2015, pp. 1613–1622.

[22] A. Graves, "Practical variational inference for neural networks," in *Advances in Neural Information Processing Systems 24*. Curran Associates, Inc., 2011, pp. 2348–2356.

[23] C. Louizos and M. Welling, "Structured and efficient variational deep learning with matrix gaussian posteriors," in *Proceedings of The 33rd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 48. New York, New York, USA: PMLR, 20–22 Jun 2016, pp. 1708–1716.

[24] J. T. Springenberg, A. Klein, S. Falkner, and F. Hutter, "Bayesian optimization with robust bayesian neural networks," in *Advances in Neural Information Processing Systems 29*. Curran Associates, Inc., 2016, pp. 4134–4142.

[25] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," in *Proceedings of the 33rd International Conference on Machine Learning (ICML-16)*, 2016.

[26] I. Osband, C. Blundell, A. Pritzel, and B. Van Roy, "Deep exploration via bootstrapped dqn," in *Advances in Neural Information Processing Systems 29*. Curran Associates, Inc., 2016, pp. 4026–4034.

[27] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[28] B. Efron and R. J. Tibshirani, *An Introduction to the Bootstrap*. New York: Chapman & Hall, 1993.

[29] G. Kahn, A. Villaflor, V. Pong, P. Abbeel, and S. Levine, "Uncertainty-aware reinforcement learning for collision avoidance," *CoRR*, vol. abs/1702.01182, 2017.

[30] N. M. Amato, S. S. Srinivasa, N. Ayanian, and S. Kuindersma, Eds., *Robotics: Science and Systems XIII, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, July 12-16, 2017*, 2017.

[31] T. Fan, P. Long, W. Liu, and J. Pan, "Fully Distributed Multi-Robot Collision Avoidance via Deep Reinforcement Learning for Safe and Efficient Navigation in Complex Scenarios," *CoRR*, Aug. 2018.

[32] J. Lygeros, C. J. Tomlin, and S. S. Sastry, "Controllers for reachability specifications for hybrid systems," *Automatica*, vol. 35, pp. 349–370, 1999.

[33] A. Majumdar and R. Tedrake, "Funnel libraries for real-time robust feedback motion planning," *CoRR*, vol. abs/1601.04037, 2016.

[34] T. J. Perkins and A. G. Barto, "Lyapunov design for safe reinforcement learning," *J. Mach. Learn. Res.*, 2003.

[35] L. Liebenwein, C. Baykal, I. Gilitschenski, S. Karaman, and D. Rus, "Sampling-based approximation algorithms for reachability analysis with provable guarantees," in *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.

[36] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" in *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc., 2017, pp. 5574–5584.

[37] N. Pawlowski, M. Rajchl, and B. Glocker, "Implicit weight uncertainty in neural networks," *CoRR*, vol. abs/1711.01297, 2017.

[38] M. Teye, H. Azizpour, and K. Smith, "Bayesian Uncertainty Estimation for Batch Normalized Deep Networks," *CoRR*, Feb. 2018.

[39] W. R. Thompson, "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples," *Biometrika*, vol. 25, pp. 285–294, 1933.

[40] Y. Liu, P. Ramachandran, Q. Liu, and J. Peng, "Stein variational policy gradient," *CoRR*, vol. abs/1704.02399, 2017.

[41] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[42] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[43] A. Vemula, K. Mülling, and J. Oh, "Social attention: Modeling attention in human crowds," *CoRR*, vol. abs/1710.04689, 2017.

[44] Z. Che, S. Purushotham, K. Cho, D. A. Sontag, and Y. Liu, "Recurrent neural networks for multivariate time series with missing values," in *Scientific Reports*, 2018.

[45] G. Lee, S. S. Srinivasa, and M. T. Mason, "Gp-ilqg: Data-driven robust optimal control for uncertain nonlinear dynamical systems," *CoRR*, vol. abs/1705.05344, 2017.

[46] E. Theodorou, Y. Tassa, and E. Todorov, "Stochastic differential dynamic programming," in *Proceedings of the 2010 American Control Conference*, 2010.

[47] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *CoRR*, vol. abs/1606.01540, 2016.

[48] J. P. van den Berg, S. J. Guy, M. C. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *ISRR*, 2009.