Predictive Risk for Efficient Black-Box Validation of Autonomous Vehicles

Robert J. Moss [1]

Abstract

We are interested in data efficient black-box validation techniques to assess autonomous vehicle (AV) risk. We falsify an AV policy in a noisy environment using a reinforcement learning technique called *adaptive stress testing*. Once we collect a supervised dataset of failures (i.e., a collision with another vehicle), we learn models to be later used in a more data efficient failure search. Given a feature input of the distance to the other agent and the closure rate, the models predict whether the current scenario will result in a failure, which we can use to augment the reward function to bias the search towards failures. We use two types of Gaussian discriminant analysis, namely *linear discriminant analysis* and *quadratic discriminant analysis*. We also compare against a support-vector machine classifier. Results suggest that including a signed additive prediction of risk to the reward function leads to much higher failure rates, thus resulting in more data efficient falsification. The learned generative models are then compared to an empirical cost distribution to perform AV policy risk assessment.

1. Motivation

To assess the risk of autonomous vehicle (AV) policies, we would like to be able to do two important things: (1) efficiently find trajectories that led to a collision with another vehicle (i.e., a failure) and (2) compute the distribution of failures over the input features x for later risk assessment. In our case, the input features consist of the distance between the ego vehicle and the other agent, and the closure rate with the other agent (i.e., a measure of the cost/severity of collision). We restrict ourselves to these input features as they are already provided given the *adaptive stress testing* (AST) formulation, thus keeping the black-box assumption (where AST requires a "miss distance" to the failure to guide its search, and the rate value is derived from this distance).

[1]Computer Science, Stanford University, Stanford, CA 94305. Correspondence to: Robert J. Moss <mossr@cs.stanford.edu>.

Final Project for Stanford University CS229 Machine Learning, Spring 2021.

To collect a large dataset of failure and non-failure AV trajectories, we will use AST (Lee et al., 2020) to control sensor noise to search for failures using reinforcement learning. The AST approach formulates the black-box validation problem as a Markov decision process (MDP) with a reward function that guides towards both failures (often called *falsification*) and high-likely failures (Corso et al., 2020). A benefit of our approach is that we can automatically collect a supervised dataset using AST as the collection method, but we could have also been given a dataset and used that to train the models instead. Given the dataset generated by AST, we discuss how to efficiently find more failure examples and how we can compute the distribution of failures to assess AV risk. We apply *generative modeling* techniques to act as a critic that predicts failures and that can compute failure distributions for later risk assessment.

2. Related Work

The problem of black-box validation of safety-critical systems has been studied extensively (Corso et al., 2020). Failure predictions in autonomous vehicles have been primarily studied to predict imminent disengagements to be used to safety pass control back to the human drivers (Hecker et al., 2018; Kuhn et al., 2020)—which differs from our proposed work to use risk predictions for more efficient falsification. Hecker et al. (2018) train a supervised approach using recurrent neural networks given video frames of impending failures. Their approach focuses on "scene drivability" to assess whether a particular scene is too complex for the AV policy to operate correctly, but requires a large amount of specified visual training data containing both normal and unsafe operations. Another similar approach from Kuhn et al. (2020) use LSTM-based models to predict future disengagements while treating the car as a black box, yet they still require state information which, under certain definitions, violates the black-box assumption. Despite this, their approach focuses on the use of the failure prediction to be integrated into the AV stack during driving, as opposed to using the prediction in simulation for validation. Actor-critic methods have been proposed for the purpose of developing safe AV policies (Gupta et al., 2020) rather than using actor-critic methods for the validation task itself.

The motivation to preserve the black-box assumption is driven by the rise of complex methods used in the devel-

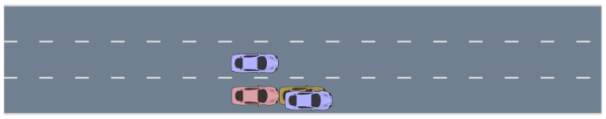


Figure 1. Validation scenario: A stopped vehicle on a three-lane highway. The red vehicle is the ego vehicle (i.e., AV policy) and the yellow vehicle is the "other" agent. The purple vehicles show the positional beliefs based on added noise disturbances. This figure is showing the terminal state when a collision occurred.

opment of AV policies, such as neural network-based controllers (Kiran et al., 2021). Strickland et al. (2018) explore a deep learning-based approach to predict AV collisions in simulation for better autonomous vehicle risk assessment. Their work keeps the black-box assumption to extend to other AV policies, yet they rely on large neural networks trained on images to predict the collisions.

There are companies focused solely on efficient black-box validation of autonomous systems, namely Trustworthy AI (Norden et al., 2020). Their validation technique uses adaptive importance sampling methods to perform an unbiased search for rare failure events in a commercial autonomous system, namely Comma AI's OPENPILOT. Although their approach is shown to be more tractable than naive Monte Carlo estimation, they use Markov chain Monte Carlo to estimate rare event probabilities which may be data inefficient when consecutive iterations are far from each other in the design space; thus, the conditional probability distributions may not be wide enough to cross over at each iteration.

Other AV validation work that extends the AST method have focused on efficiently finding likely failures (Koren & Kochenderfer, 2019) and diversifying the types of failures found (Corso et al., 2019). Koren & Kochenderfer (2019) apply AST to the validation of an AV policy in simulation to improve efficiency in black-box validation. They employ a recurrent neural network-based solution to the AST problem formulation to find high-likely failures more efficiently while still maintaining the black-box assumption. Their focus is more on the particular *solvers* used to find failures, while we propose an approach that augments the MDP *reward function* itself (used by all of the solvers), which potentially allows our method to be broadly applied to all types of solvers to achieve more data efficient falsification.

3. Dataset and Features

We consider the scenario of a stopped vehicle on a three-lane highway with an approaching ego vehicle from behind, as illustrated in fig. 1. AST acts as an adversary and controls the noise disturbances applied per time step to try and find failures (Koren et al., 2019). Because we are operating in a reinforcement learning setting, we treat training and test datasets in a different manner. We run the standard AST failure search (i.e., without any risk prediction) across 10 different RNG seeds to better assess the performance and to gauge the failure rate and the highest likelihood of failure with accompanying statistics. At the end of each simulation, sometimes referred to as the *terminal state*, we collect the distance d from the ego vehicle to the other vehicle and the closure rate r (which is derived from the distance). We use these two measurements as our input features $x = [r, d]$ in our models. We collect the supervised training dataset by storing the features x measured at the terminal state and the target y which is a binary value indicating a collision occurred (i.e., a failure). We define the terminal state as either when a collision occurs or when the simulation ends (after 30 seconds). Note that we arbitrarily choose the final seed to start our data collection for the training set; this way we have a dataset representative of a single standard AST failure search and not one that is a combination across many simulations. Explained further in section 5, we also do this to show that a useful model can be learned from a small amount of failure data points.

For testing, we select 10 *different* RNG seeds than those used in training—simply to remove any unintentional data leakage in how the vehicle trajectories were sampled in simulation. We employ the predictive risk estimate in the reward function during these simulations and collect relevant statistics for performance assessment, described in section 5. It is also worth noting that we are using the Monte Carlo tree search (MCTS) algorithm as our solver, which uses stochasticity (Coulom, 2006), but emphasize that our proposed method is agnostic to the particular type of solver due to simply being an augmentation of the reward function.

4. Method

The main approach is illustrated in fig. 2. The first phase (COLLECTION) runs the standard AST failure search and collects the features x at the termination state into a dataset $\mathcal{D}$. Using this supervised dataset, the next phase (LEARNING) trains a model to act as a critic $\mathcal{C}$ to predict failures during future simulations (providing a signed magnitude value of how close the input features are to the decision boundary). The final phase (EFFICIENT SEARCH) uses the critic to augment the AST reward function by including an additive penalty/reward based on the failure prediction (where positive values indicate failure predictions). Lastly, a failure distribution $\mathcal{F}$ is output and used for risk assessment.

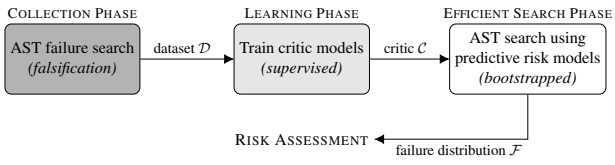


Figure 2. Phased efficient validation using predictive risk.

The use of generative models gives us the flexibility to compute the distribution $p(x \mid y = 1)$ where $y = 1$ indicates a failure/collision occurred. Using different versions of *Gaussian discriminant analysis* (GDA) (McLachlan, 1992), we can model the outcome $y \sim \text{Bernoulli}(\phi)$ with unknown failure rate ϕ and the individual conditional distributions

$$x \mid y = 0 \sim \mathcal{N}(\mu_0, \Sigma_0) \tag{1}$$
$$x \mid y = 1 \sim \mathcal{N}(\mu_1, \Sigma_1) \tag{2}$$

as multivariate Gaussians. We can then solve for the optimal parameters using maximum likelihood estimation (MLE). We also use these models as discriminators to predict a signed value of "closeness" to failure.

4.1. Linear discriminant analysis (LDA)

We predict a failure given a distance d to the other agent (i.e., "how close are we to fail") and a closure rate r (i.e., simply the change in distance over time: $r_t = d_{t-1} - d_t$). We concatenate these features into our input $x = [r, d]$. If we (incorrectly) assume the covariances are the same (i.e., $\Sigma = \Sigma_0 = \Sigma_1$), then we get *linear discriminant analysis* (Ghojogh & Crowley, 2019):

$$\delta_k(x) = x^\top \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k \Sigma^{-1} \mu_k + \log(\pi_k) \tag{3}$$

and when $\delta_0(x) < \delta_1(x)$, we classify as *failure* (i.e., class 1). This gives us a "hard" decision boundary for classification as seen in fig. 3(a). Note that π_k is the prior on the failure rate, where we set $\pi_k = 0.5$ (i.e., a uniform prior). Setting $\delta_0(x) = \delta_1(x)$ and solving for $\delta(x) = 0$, we get:

$$\begin{aligned}\delta(x) =& (x - \mu_0)^\top \Sigma^{-1} (x - \mu_0) \\ & - (x - \mu_1)^\top \Sigma^{-1} (x - \mu_1)\end{aligned} \tag{4}$$

and if $\delta(x) > 0$, then we classify as *failure*. This gives us a "soft" decision boundary for classification (fig. 3(b)).

In our case, notice that when using LDA (fig. 3(a) and fig. 3(b)), the assumption of a shared covariance does not hold. Notably, the distribution of failure events has a thinner spread than the non-failure events. Thus, we investigate the case where the covariance matrices can be different which leads us to *quadratic discriminant analysis*.

4.2. Quadratic discriminant analysis (QDA)

Quadratic discriminant analysis (Hastie et al., 2001), gives:

$$\begin{aligned}\delta_k(x) = & -\frac{1}{2} \mu_k \Sigma_k^{-1} \mu_k + x^\top \Sigma_k^{-1} \mu_k \\ & - \frac{1}{2} x^\top \Sigma_k^{-1} x - \frac{1}{2} \log |\Sigma_k|\end{aligned} \tag{5}$$

with separate covariances Σ_k, and when $\delta_0(x) < \delta_1(x)$, we classify as *failure* (i.e., class 1). This gives us a "hard" decision boundary as seen in fig. 3(c). Setting $\delta_0(x) = \delta_1(x)$ and solving for $\delta(x) = 0$, we get:

$$\begin{aligned}\delta(x) =& (x - \mu_0)^\top \Sigma_0^{-1} (x - \mu_0) + \log |\Sigma_0| \\ & - (x - \mu_1)^\top \Sigma_1^{-1} (x - \mu_1) - \log |\Sigma_1|\end{aligned} \tag{6}$$

and when $\delta(x) > 0$, we classify as *failure* (i.e., class 1) to give us a "soft" decision boundary as seen in fig. 3(d). Notice that QDA provides a better fit, and results in table 1 suggest it generally outperforms LDA.

The standard AST reward function $R(s_t, a_t)$ for a state s_t consisting of a collection of all noise samples up to time t and action a_t of the noise itself, is described as follows:

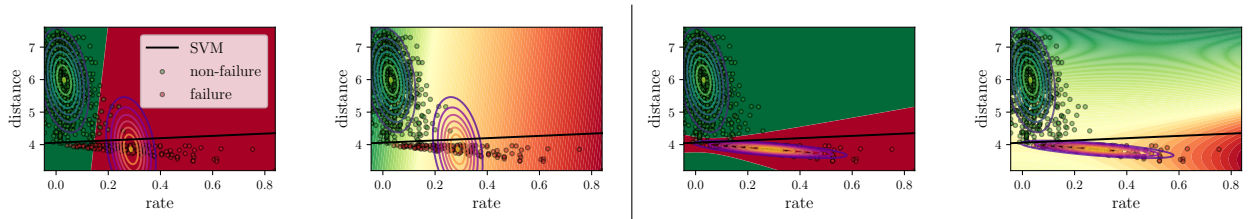
$$R(s, a) = \begin{cases} 0 & \text{if } s_T \in E \\ -d(s) & \text{if } s_T \notin E \\ \log p(a \mid s) & \text{otherwise} \end{cases} \tag{7}$$

where $d(s)$ is the "distance to failure" metric, $\log p(a \mid s)$ is the log-likelihood of the noise disturbance, s_T indicates the terminal state, and E is the set of failure events. Thus, this reward function guides the search towards failures by minimizing $d(s)$ when no failure is found, while maximizing the sum of the log-probabilities to find *likely* failures (which is equivalent to maximizing the product of the probabilities). We then take the failure prediction function $\delta(x)$ (described in eq. (4) for LDA and eq. (6) for QDA) and use the signed value given by $\delta(x)$ as a penalty/reward based on the predictive risk to create the augmented reward function $R'(s, a, x) = R(s, a) + \delta(x)$. Negative values of $\delta(x)$ indicate a non-failure prediction (penalty) where positive values of $\delta(x)$ indicate a failure prediction (reward).

5. Experiments, Results, and Discussion

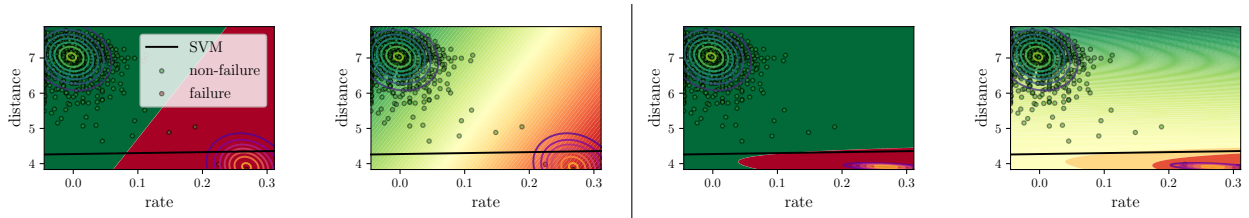
Experiments were designed to test the performance of our approach in a "non-restrictive" noisy scenario where failures are common (around a nominal 14% failure rate, illustrated in fig. 3) and in a "restrictive" noisy scenario where failures are rare (around a 0.4% failure rate, illustrated in fig. 4). We do this to emphasize the effectiveness of our approach when very little failure data is available during training. The "non-restrictive" noisy scenario has xy-position disturbances sampled from a zero-mean Gaussian with $\sigma = 3$, and the "restrictive" noisy scenario sets $\sigma = 2$.

The GDA approaches are compared to an SVM classifier as a baseline. Seeing that SVM predicts $\delta(x) = \hat{y} \in \{-1, 1\}$, while the "soft" LDA/QDA provide a sign *and* magnitude, we are interested in scaling factors C to hopefully achieve better performance of SVM using $C\delta(x)$ instead. Similarly, the "hard" decision boundary of LDA (fig. 3(a)) and QDA (fig. 3(c)) are also tested using different scale factors. We swept values of C and a scale factor of $C = 10{,}000$ performs well across all three algorithms; thus, we will use this



(a) LDA: "Hard" boundary. (b) LDA: "Soft" boundary. (c) QDA: "Hard" boundary. (d) QDA: "Soft" boundary.

Figure 3. Non-restrictive: Decision boundaries using LDA and QDA, including multivariate Gaussian fits and SVM boundary.



(a) LDA: "Hard" boundary. (b) LDA: "Soft" boundary. (c) QDA: "Hard" boundary. (d) QDA: "Soft" boundary.

Figure 4. Restrictive: Decision boundaries using LDA and QDA, including multivariate Gaussian fits and SVM boundary.

value throughout the analysis. Note that this is a major drawback of the SVM and "hard" approaches as we have to tune C accordingly. Table 1 also compares against a random prediction of $\hat{y} \in \{-C, C\}$ and surprisingly has higher failure rate than nominal in the non-restrictive case (yet with large variance and poor training performance) and lower in the restrictive case, but the predictive models still significantly outperform random. The increase in failure rate for the random approach could be attributed to the non-restrictive case being more likely to find failures (thus, random large rewards/penalties could help exploration), which is confirmed when comparing to random in the restrictive case.

We collected a supervised dataset $\mathcal{D}$ of 1000 AV trajectories over 10 different RNG seeds (for statistical significance and reproducibility) and we arbitrarily select the collected data from the final run to train our predictive model. The driving scenario is shown in fig. 1 and we use the `AutomotiveSimulator.jl`[1] Julia package for simulations, `POMDPStressTesting.jl`[2] for AST (Moss, 2021), and the *intelligent driver model* (IDM) from Treiber et al. (2000) as the AV policy we are validating. We test the performance of our model through the EFFICIENT SEARCH PHASE using 10 *different* RNG seeds than were used for training/baselining. Finally, we perform risk assessment using the failure distribution $\mathcal{F}$ learned by the generative models, comparing to the empirical cumulative distribution function (eCDF) computed directly from data.

Figure 5 shows a proxy for learning to find failures in simulation (where we do not compare rewards or returns because the predictive approach augments the reward function itself). Comparing the QDA "soft" approach against standard AST, fig. 5 suggests that we are more effectively minimizing the distance between the two vehicles, which we would expect to translate into more failures. Table 1 shows that this assumption is correct as the predictive approaches find about 3-5 times more failures in the "non-restrictive" case and about 28-38 times more failures in the "restrictive" case, all using the same number of episodes (with negligible computational cost). Notice that we learn an effective model in fig. 4 with only three failure data points.

[1] https://github.com/sisl/AutomotiveSimulator.jl

[2] https://github.com/sisl/POMDPStressTesting.jl

5.1. Error Analysis

Figures 6(a) and 6(b) show the confusion matrices for the two test cases using the QDA "soft" approach (refer to table 1 for performance metrics). We compute these matrices over all 10,000 episodes (i.e., 10 seeds $\times$ 1000 episodes). These figures suggest that the learned models perform well. This is also evident in the increased failure rate seen in table 1. We include the standard deviations of each metric across the RNG seeds to indicate the potential noisiness in the performance and report the values computed on the training dataset in parentheses. Results show the predictive approaches outperform the nominal case across all metrics.

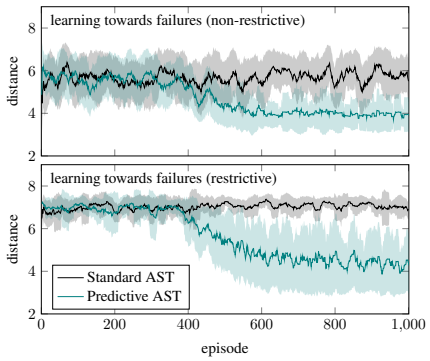
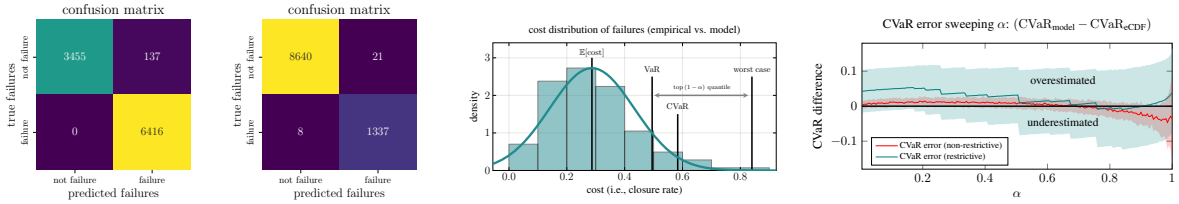


Figure 5. Miss distance at episode termination (proxy for learning).

Table 1. Performance metrics: Testing and (training).

EXPERIMENT	APPROACH	FAIL-RATE*	$\max \log(p)$*	PRECISION*	RECALL*	ACCURACY*
	NOMINAL	$0.14_{\pm 0.01}$	$8.15_{\pm 0.09}$	—	—	—
	RANDOM†	$0.21_{\pm 0.17}$ (0.086)	$8.19_{\pm 0.17}$ (7.87)	$0.215_{\pm 0.168}$ (0.149)	$0.511_{\pm 0.054}$ (0.524)	$0.502_{\pm 0.021}$ (0.504)
"NON-RESTR."	QDA "SOFT"	$\mathbf{0.64_{\pm 0.11}}$ (0.60)	$8.26_{\pm 0.07}$ (8.24)	$0.978_{\pm 0.011}$ (0.899)	$\mathbf{1.0_{\pm 0.0}}$ (1.0)	$0.986_{\pm 0.006}$ (0.984)
	QDA "HARD"‡	$0.61_{\pm 0.14}$ (0.77)	$8.28_{\pm 0.07}$ (8.38)	$0.973_{\pm 0.014}$ (0.899)	$\mathbf{1.0_{\pm 0.0}}$ (1.0)	$0.985_{\pm 0.005}$ (0.984)
	LDA "SOFT"	$0.58_{\pm 0.11}$ (0.75)	$8.25_{\pm 0.11}$ (8.29)	$0.982_{\pm 0.009}$ (0.900)	$0.965_{\pm 0.013}$ (0.818)	$0.971_{\pm 0.007}$ (0.961)
	LDA "HARD"‡	$0.44_{\pm 0.15}$ (0.54)	$\mathbf{8.30_{\pm 0.08}}$ (8.43)	$0.972_{\pm 0.019}$ (0.900)	$0.875_{\pm 0.024}$ (0.818)	$0.938_{\pm 0.013}$ (0.961)
	SVM‡	$0.64_{\pm 0.11}$ (0.67)	$8.28_{\pm 0.08}$ (8.23)	$\mathbf{0.992_{\pm 0.004}}$ (0.960)	$\mathbf{1.0_{\pm 0.0}}$ (1.0)	$\mathbf{0.995_{\pm 0.002}}$ (0.994)
	NOMINAL	$0.004_{\pm 0.002}$	$8.58_{\pm 0.74}$	—	—	—
	RANDOM†	$0.002_{\pm 0.001}$ (0.003)	$8.26_{\pm 1.13}$ (8.45)	$0.003_{\pm 0.002}$ (0.004)	$0.642_{\pm 0.393}$ (0.667)	$0.496_{\pm 0.015}$ (0.487)
"RESTRICTIVE"	QDA "SOFT"	$0.13_{\pm 0.14}$ (0.38)	$\mathbf{9.35_{\pm 0.73}}$ (9.96)	$\mathbf{0.972_{\pm 0.050}}$ (1.0)	$0.997_{\pm 0.006}$ (1.0)	$0.997_{\pm 0.003}$ (1.0)
	QDA "HARD"‡	$0.15_{\pm 0.20}$ (0.47)	$9.23_{\pm 0.59}$ (9.74)	$0.932_{\pm 0.140}$ (1.0)	$0.999_{\pm 0.002}$ (1.0)	$\mathbf{0.998_{\pm 0.002}}$ (1.0)
	LDA "SOFT"	$\mathbf{0.15_{\pm 0.20}}$ (0.40)	$9.26_{\pm 0.93}$ (9.89)	$0.911_{\pm 0.157}$ (0.600)	$0.930_{\pm 0.210}$ (1.0)	$0.997_{\pm 0.003}$ (0.998)
	LDA "HARD"‡	$0.11_{\pm 0.14}$ (0.11)	$9.07_{\pm 0.80}$ (9.81)	$0.830_{\pm 0.179}$ (0.600)	$0.968_{\pm 0.068}$ (1.0)	$0.994_{\pm 0.005}$ (0.998)
	SVM‡	$0.13_{\pm 0.18}$ (0.48)	$9.03_{\pm 0.99}$ (10.00)	$0.885_{\pm 0.198}$ (1.0)	$\mathbf{1.0_{\pm 0.0}}$ (1.0)	$0.996_{\pm 0.003}$ (1.0)

* Training results shown in parentheses. † Random prediction of $\hat{y} \in \{-C, C\}$. ‡ With scale factor $C = 10{,}000$.



(a) Non-restrictive. (b) Restrictive. (c) Model/empirical cost distr. (d) CVaR error in model.

Figure 6. Confusion matrices, the empirical cost distribution compared to the model, and the modeled CVaR error.

5.2. Risk Assessment

The use of generative models not only gives us predictions we can use for efficient falsification but also provides a distributional model $\mathcal{F}$ of the *cost* of failure (where we use the closure rate at time of collision as a measure of cost). We isolate the univariate Gaussian of the rate dimension r conditioned on $y = 1$ from eq. (2) to get the distributional model $\mathcal{F}(r) := \mathcal{N}\big(r \mid \mu_1^{(1)}, \Sigma_1^{(11)}\big)$, where the superscripts are the indices of the rate component. We use this cost model to derive risk metrics associated with failures and compare it to the empirical cumulative distribution function (eCDF) as shown in fig. 6(c). Common risk metrics used in the financial and robotics communities are expected cost, *value at risk* (VaR), *conditional value at risk* (CVaR), and the worst case cost (Majumdar & Pavone, 2017). The industries have shifted their primary focus to use CVaR as a measure of risk as it can be shown to be the most robust (Majumdar & Pavone, 2017). CVaR is defined as the expected cost under the conditional distribution set by a *risk tolerance* α (where we are willing to accept α fraction of the highest risk). Sweeping α, we compute CVaR from the eCDF and from the learned QDA "soft" model $\mathcal{F}$ sampled 1000 times, then compare their difference in fig. 6(d), where values above zero are an overestimation from the model. Notice that both scenarios accurately capture CVaR, where the mean $\text{CVaR}_{\text{eCDF}}$ across values of α is about 0.417 with a mean error of 0.0006 (i.e., an error of 0.14% for the non-restrictive case) and the mean $\text{CVaR}_{\text{eCDF}}$ is about 0.276 with a mean error of 0.024 (i.e., an error of 8.7% for the restrictive case). Thus, the models could be used to further assess AV risk.

6. Conclusions and Future Work

The simplicity of the learned models is a major benefit of this predictive approach. We are able to more efficiently find failures, even when minimal failure data is present in the initial simulations. The addition of a signed risk prediction measurement in the AST reward function serves as another way to guide the search towards likely failures. We use input features of distance and rate that are available to us in simulation, thus keeping the AST black-box assumption; allowing this approach to extend to other systems to more efficiently falsify. Through our experiments, we have shown that the predictive methods can increase failure rate by about 3-38 times relative to the nominal AST failure search. SVMs are also shown to be effective, but the GDA-based generative models can be used for both prediction and risk assessment by outputting a cost distribution of failures.

Future work could investigate the input features to see if both distance and rate are necessary. Omitted from this work for brevity is the investigation of using the *full trajectory* of features instead of only collected at the terminal state. Preliminary results suggest that collecting features at the terminal state provides higher failure rates in most cases, but further work could explore ways to incorporate temporal features throughout the simulation. Fitting a Gamma distribution to the rate at collision may also provide better performance in risk estimation simply due to observations that the cost distribution closely follows a Gamma distribution. We would also like to test this approach across many different driving scenarios to see how well it generalizes.

References

Corso, A., Du, P., Driggs-Campbell, K., and Kochenderfer, M. J. Adaptive stress testing with reward augmentation for autonomous vehicle validatio. In *Intelligent Transportation Systems Conference (ITSC)*, pp. 163–168, 2019. doi: 10.1109/ITSC.2019.8917242.

Corso, A., Moss, R. J., Koren, M., Lee, R., and Kochenderfer, M. J. A survey of algorithms for black-box safety validation, 2020. arXiv: 2005.02979.

Coulom, R. Efficient selectivity and backup operators in monte-carlo tree search. In *International conference on computers and games*, pp. 72–83. Springer, 2006.

Ghojogh, B. and Crowley, M. Linear and quadratic discriminant analysis: Tutorial, 2019. arXiv: 1906.02590.

Gupta, A., Khwaja, A. S., Anpalagan, A., Guan, L., and Venkatesh, B. Policy-gradient and actor-critic based state representation learning for safe driving of autonomous vehicles. *Sensors*, 20(21), 2020. ISSN 1424-8220. doi: 10.3390/s20215991.

Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning*. Springer, 2001.

Hecker, S., Dai, D., and Van Gool, L. Failure prediction for autonomous driving. In *Intelligent Vehicles Symposium (IV)*, pp. 1792–1799, 2018. doi: 10.1109/IVS.2018.8500495.

Kiran, B. R., Sobh, I., Talpaert, V., Mannion, P., Sallab, A. A. A., Yogamani, S., and Pérez, P. Deep reinforcement learning for autonomous driving: A survey. *Transactions on Intelligent Transportation Systems*, pp. 1–18, 2021.

Koren, M. and Kochenderfer, M. J. Efficient autonomy validation in simulation with adaptive stress testing. In *Intelligent Transportation Systems Conference (ITSC)*, pp. 4178–4183, 2019. doi: 10.1109/ITSC.2019.8917403.

Koren, M., Corso, A., and Kochenderfer, M. J. The adaptive stress testing formulation. In *Workshop on Safe Autonomy, Robotics: Science and Systems*, 2019.

Kuhn, C. B., Hofbauer, M., Petrovic, G., and Steinbach, E. Introspective black box failure prediction for autonomous driving. In *Intelligent Vehicles Symposium (IV)*, pp. 1907–1913, 2020. doi: 10.1109/IV47402.2020.9304844.

Lee, R., Mengshoel, O. J., Saksena, A., Gardner, R., Genin, D., Silbermann, J., Owen, M., and Kochenderfer, M. J. Adaptive stress testing: Finding likely failure events with reinforcement learning. *Journal of Artificial Intelligence Research*, 69:1165–1201, 2020.

Majumdar, A. and Pavone, M. How should a robot assess risk? towards an axiomatic theory of risk in robotics, 2017. arXiv: 1710.11040.

McLachlan, G. J. Discriminant analysis and statistical pattern recognition. *Wiley Series in Probability and Mathematical Statistics*, 1992.

Moss, R. J. POMDPStressTesting.jl: Adaptive stress testing for black-box systems. *Journal of Open Source Software*, 6(60):2749, 2021. doi: 10.21105/joss.02749.

Norden, J., O'Kelly, M., and Sinha, A. Efficient black-box assessment of autonomous vehicle safety, 2020. arXiv: 1912.03618.

Strickland, M., Fainekos, G., and Amor, H. B. Deep predictive models for collision risk assessment in autonomous driving. In *International Conference on Robotics and Automation (ICRA)*, pp. 4685–4692, 2018. doi: 10.1109/ICRA.2018.8461160.

Treiber, M., Hennecke, A., and Helbing, D. Congested traffic states in empirical observations and microscopic simulations. *Physical Review E*, 62(2):1805, 2000.