Predictive Monitoring of Traffic Rules

Luis Gressenbuch and Matthias Althoff

Abstract— **Predicting the trajectories of other road users relies to a large extent on the assumption that they adhere to the legally binding traffic rules. Hence, when this assumption does not hold anymore, the prediction becomes invalid, putting autonomous vehicles relying on such predictions in a critical situation. We propose a solution to this problem by predicting traffic rule violations. All traffic rules are modeled by temporal logic, and we provide real-valued generalizations of required logical predicates to obtain features for prediction with neural networks. The usefulness of our approach is demonstrated by predicting rule violations on a dataset recorded from a highway. Our results show that directly learning traffic rule violations using the features from temporal logic formulas often performs better compared to separately predicting and monitoring trajectories.**

I. Introduction

Incorporating traffic rules into autonomous driving systems has become a focus of recent research. This trend does not only help vehicles drive in compliance with traffic rules but also improves the predictability of other traffic participants [1]. For instance, another vehicle will most likely slow down in front of an intersection if it does not have the right of way. Observations of illegal behavior have been previously considered by adjusting or removing the respective predictions [1] or constraints [2], [3] once a violation has been observed. While these approaches are suitable for traffic rule violations such as speeding, they could be insufficient in more critical scenarios, where the observation of a rule violation comes too late, e.g., running a red light at an intersection or a pedestrian unexpectedly crossing the road.

To prevent severe consequences, traffic rule violations in highly critical scenarios must be explicitly predicted. Advances in two areas help to realize this goal: Machine learning techniques have produced remarkable results for predicting trajectories of surrounding vehicles [4], [5]. Furthermore, recent attempts to formalize traffic rules in temporal logic have enabled automated traffic rule monitoring [6], [7]. However, research has not specifically targeted the prediction of traffic rule violations so far. Major challenges for this task remain as critical scenarios are often inherently difficult to foresee, occur only rarely, and are thus infrequently contained in recorded datasets.

A. Related Work

The foundations of predictive monitoring of traffic rules consist of two major research directions: Formalization of traffic rules in temporal logic and predictive monitoring of temporal logic specifications, which we both present subsequently.

1) Formalization of Traffic Rules: Incorporating traffic rules in any kind of computation, e.g., perception, prediction, planning, etc., first requires their formalization in a machine-readable manner [8]. Most approaches have identified temporal logic as a suitable specification formalism. Linear temporal logic (LTL) [9] is based on propositional logic and temporal operators, such as *always, until, next*, and *eventually*. It has been used for specifying motion [10] and formalizing traffic rules [6]. To overcome the limitations of a discrete notion of time in LTL, metric temporal logic (MTL) [11] has been used to formalize a comprehensive set of highway traffic rules [7]. While LTL and MTL only provide Boolean values for satisfaction and violation of a property, signal temporal logic (STL) [12] with quantitative semantics [13] allows one to quantify the degree of satisfaction or violation of a property, also known as *robustness degree*. Hekmatnejad et al. [14] have formalized the safe distance rules of the *responsibility sensitive safety* framework [15] in STL, while Cho et al. [16] and Li et al. [17] have specified basic traffic rules, e.g., collision avoidance, lane-keeping, or slowly approaching a pedestrian crossing.

2) Predictive Monitoring: Predicting compliance with defined temporal properties is an important task in different domains. Quin and Deshmukh [18] used autoregressive integrated moving average models to predict the probability of satisfying an STL specification by fitting them to a time series. They demonstrate their method using specifications for various applications including automated insulin delivery and control of an unmanned aerial vehicle. However, the method requires the predicted quantity to be modeled as an autoregressive integrated moving average process, which can only model temporally correlated time series. Thus, series-spatial correlations, as found in traffic rules [19], cannot be incorporated. Ma et al. [20] proposed the predictive monitoring approach *CityPM* for STL specifications in smart cities. They extended STL to express uncertainty and first predict the signals using Bayesian recurrent neural networks and afterward check satisfaction of the STL specifications with monitors.

To the best of our knowledge, the first work that incorporates a prediction of traffic rule conformance is presented by Cho et al. [16]. State sequences of vehicles are used as features for a recurrent neural network (RNN) predicting the probability distribution of robustness values of STL formulas. The robustness values are used to eliminate samples of predicted trajectories not conforming to the predicted robust-

All authors are with the Department of Informatics, Technical University of Munich, 85748 Garching, Germany.
luis.gressenbuch@tum.de, althoff@in.tum.de

ness values. While their approach explicitly considers STL specifications, the authors only present results for simple rules and do not explicitly target the prediction of traffic rule violations. Li et al. [17] have leveraged existing trajectory prediction algorithms with *syntax tree features* from quantitative semantics of STL within a generative adversarial network to create trajectory predictions aware of traffic rules. The approach was trained on a dataset with $20\,\%$ of the samples violating traffic rules. While the robustness features used in this approach are similar to ours, only one STL formula with the always operator has been investigated. Moreover, the performance of accurately predicting rule violations has not been evaluated.

Several general and highway-specific traffic rules have been formalized in LTL and MTL [6], [7]. Only few basic rules were formalized in STL [14], [16], [17]. However, traffic rules formalized in STL are widely applicable as they provide a quantitative measure of rule compliance. Existing approaches for predictive monitoring of STL specifications have mostly split the task into a prediction and monitoring task [18], [20]. Approaches that fuse these two tasks were proposed only for predicting trajectories, but not for predicting traffic rule violations [16], [17]. By combining prediction and monitoring, one can optimize the specific purpose of foreseeing traffic rule violations. Furthermore, behavior specifications or basic traffic rules for which many violating training examples exist have been considered.

B. Contributions

We propose a methodology to predict violations of traffic rules modeled using temporal logic formulas and specifically contribute the following novelties:

- We provide an approach for predictive monitoring of temporal logic specifications using neural networks;
- we define the robustness degree of predicates for the quantitative evaluation of traffic rules in STL; and
- we evaluate the proposed approach on real-world data using two types of neural networks and compare the results with a trajectory prediction approach.

The rest of the paper is organized as follows: The theoretical foundations are introduced in Section II. In Section III, we present our approach for predicting violations of traffic rules. In Section IV, the neural networks are defined, and we evaluate our approach on a dataset from highway traffic. Section V discusses our results.

II. Preliminaries

A. Signals

We use the STL standard definition of signals as the basis for our work [21, Sec. 2.1] and discrete time steps to resemble a sampled perception of the environment. A signal $w = w_0 w_1 \cdots$ is a sequence of vectors $w_k \in \mathbb{R}^r$. We write $w[k]$ for the signal vector at time step k, $w[k, k']$ for a subsequence of w from time step k to k', $\begin{pmatrix} w & w' \end{pmatrix}^\intercal = \begin{pmatrix} w[0] & w'[0] \end{pmatrix}^\intercal \begin{pmatrix} w[1] & w'[1] \end{pmatrix}^\intercal \cdots$ for a signal of concatenated signals and $\mathrm{proj}_{w'}\left(\begin{pmatrix} w & w' \end{pmatrix}^\intercal\right) = w'$ for projecting a concatenated signal to component w'. We define the finite index set $\mathcal{I} = \{1, ..., m\}$ for the m vehicles present in a scenario and x_p as the state of the vehicle with index p. For the index set $\mathcal{I} \backslash \{p\}$ without the p-th vehicle, we write $\mathcal{I}_{\neg p}$. The signal of a scenario $w_\mathcal{I} = \begin{pmatrix} x_1 & \cdots & x_m \end{pmatrix}^\intercal$ is defined as the concatenation of the signals of all vehicles.

B. Signal Temporal Logic

For the formalization of traffic rules, we use the past fragment of STL [21, Sec. 2.1], which only reasons about past signal values. Let I be a time interval of the form $[k, k']$ or $[k, \infty)$ with $k, k' \in \mathbb{N}_0$ and g an arbitrary real-valued function. The syntax of an STL formula φ is

$$\varphi ::= g(x) \geq 0 \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi \,\mathbf{S}_I\, \varphi \mid \top\,.$$

We denote $w_\mathcal{I}$ complying with φ at time step k with $w_\mathcal{I}, k \models \varphi$, and the set of predicates used in φ with $\mathcal{P}_\varphi$. We refer to [21, Sec. 2.1] for the definition of the semantics of STL and define the following constants and operators for convenience [21, Sec. 2.1], [22, Sec. 2.2]:

$$\begin{aligned} \bot &\equiv \neg\top & \mathbf{O}_I\, \varphi &\equiv \top\, \mathbf{S}_I\, \varphi \\ \varphi_1 \wedge \varphi_2 &\equiv \neg(\neg\varphi_1 \vee \neg\varphi_2) & \mathbf{P}\, \varphi &\equiv \bot\, \mathbf{S}_{[0,\infty)}\, \varphi \\ \varphi_1 \Rightarrow \varphi_2 &\equiv \neg\varphi_1 \vee \varphi_2 \end{aligned}$$

where $\mathbf{P}$ and $\mathbf{O}_I$ are untimed *previous* and timed *once* operator. Informally speaking, $\mathbf{P}\,\varphi$ holds, if φ was satisfied in the previous time step, and $\mathbf{O}_I\,\varphi$ holds, if φ was satisfied at least once in the past time interval I.

Quantitative semantics for STL has been defined to measure the robustness degree, i.e., the distance to satisfaction or violation of an STL formula. Based on the following equivalences, similar as in [21, Sec. 2.2], the robustness degree $\rho(\varphi, w_\mathcal{I}, k)$ of a signal $w_\mathcal{I}$, at time step k according to formula φ can be recursively computed:

$$\begin{aligned} \rho(\top, w_\mathcal{I}, k) &= \infty \\ \rho(\neg\varphi, w_\mathcal{I}, k) &= -\rho(\varphi, w_\mathcal{I}, k) \\ \rho(g(x) \geq 0, w_\mathcal{I}, k) &= g(\mathrm{proj}_x\,(w_\mathcal{I})\,[k]) \\ \rho(\varphi_1 \vee \varphi_2, w_\mathcal{I}, k) &= \max\{\rho(\varphi_1, w_\mathcal{I}, k), \\ &\qquad \rho(\varphi_2, w_\mathcal{I}, k)\} \\ \rho(\varphi_1\, \mathbf{S}_I\, \varphi_2, w_\mathcal{I}, k) &= \max_{k' \in (k-I) \cap \mathbb{N}_0} \min \Big\{ \rho(\varphi_2, w_\mathcal{I}, k'), \\ &\qquad \min_{k'' \in [k'+1, k-1]} \rho(\varphi_1, w_\mathcal{I}, k'') \Big\}. \end{aligned}$$

The satisfaction of φ, with respect to $w_\mathcal{I}$ is related to the sign of the robustness degree:

$$w_\mathcal{I}, k \models \varphi \iff \rho(\varphi, w_\mathcal{I}, k) \geq 0.$$

C. Environment Model

The state vector of a vehicle is defined as $\begin{pmatrix} s & d & v & a & \theta \end{pmatrix}^\intercal \in \mathbb{R}^5$, where s and d are the corresponding longitudinal and lateral position in curvilinear coordinates [23], v is the speed, a is the acceleration, and θ is the orientation. The static map is modeled as lanelet network $\mathcal{L}$, see [24]. We adopt the following operators from [7]: $\mathrm{lanes}(\cdot)$ for the lanes occupied by a vehicle, $\mathcal{O}(\cdot)$

for the occupancy of an object, $\mathrm{l}_{\{\mathrm{lb,rb}\}}(\cdot)$ for the respective left and right lane boundary, and $\mathrm{front}(\cdot)$ as well as $\mathrm{rear}(\cdot)$ for the longitudinal position of the respective frontmost and rearmost point of a vehicle. Furthermore, we define:

- The set of lanes containing the reference point of a vehicle:

$$\mathrm{ref\_lane}(x_i) = \{l \in \mathrm{lanes}(x_i) | \mathcal{O}(l) \cap \{\mathrm{proj}_{s,d}(x_j)\} \neq \emptyset\},$$

- a sign function based on set membership:

$$\mathrm{sgn}(o, \mathcal{A}) = \begin{cases} 1, & \text{if } o \in \mathcal{A} \\ -1, & \text{otherwise,} \end{cases}$$

- the distance between an element and a set:

$$\mathrm{d}(o, \mathcal{A}) = \min_{a \in \partial \mathcal{A}} \|o - a\|_2 ,$$

- the half-planes left and right of an oriented line b using the lateral component d_b of the corresponding curvilinear coordinate system [23]:

$$\mathcal{B}_\mathrm{l} = \{o \in \mathcal{O}(\mathcal{L}) | \, \mathrm{d}_b(o) \geq 0\}$$
$$\mathcal{B}_\mathrm{r} = \{o \in \mathcal{O}(\mathcal{L}) | \, \mathrm{d}_b(o) \leq 0\},$$

- the signed version of the distance and the directed Hausdorff distance to the left lane boundary:

$$\mathrm{d}_\mathrm{l}(o, b) = \mathrm{sgn}(o, \mathcal{B}_\mathrm{r}(b)) \cdot \mathrm{d}(o, b)$$
$$\mathrm{d}_{\vec{H},\mathrm{l}}(\mathcal{A}, b) = \max_{o \in \mathcal{A}} \mathrm{d}_\mathrm{l}(o, b)$$

and d_r and $\mathrm{d}_{\vec{H},\mathrm{r}}$ to the right lane boundary analogously.

D. Traffic Rules

Following the scheme of *codification* and *concretization* [25], we base our work on the existing codification of traffic rules for German highways in MTL [7]. However, we reformulate the concretization of the predicates in STL to obtain their robustness degree. Table I shows the codifications of the following traffic rules: Keeping a safe distance to the preceding vehicle (G1), avoiding unnecessary braking G2* (modified version of rule G2; see Appendix for further details) and adhering to the speed limit (G3).

TABLE I: General traffic rules from [7] with modified rule G2.

Rule	Definition of φ
G1	$\forall q \in \mathcal{I}_{\neg p}$: in-same-lane$(x_p, x_q) \wedge$ in-front-of(x_p, x_q) $\wedge \neg \mathbf{O}_{[0,t_c]}$(cut-in$(x_q, x_p) \wedge \mathbf{P}(\neg$cut-in$(x_q, x_p)))$ $\Rightarrow$ keeps-safe-distance-prec(x_p, x_q)
G2*	brakes-abruptly$(x_p) \Rightarrow \exists q \in \mathcal{I}_{\neg p}$: precedes(x_p, x_q) $\wedge$ ($\neg$keeps-safe-distance-prec(x_p, x_q) $\vee \neg$brakes-abruptly-relative(x_p, x_q))
G3	keeps-lane-speed-limit$(x_p) \wedge$ keeps-fov-speed-limit(x_p) $\wedge$ keeps-type-speed-limit$(x_p) \wedge$ keeps-brake-speed-limit(x_p)

III. Approach

Using neural networks, we combine prediction and monitoring. We start by defining our problem, followed by an overview of our approach. Afterward, we describe the computation of the robustness degree for the traffic rules defined in Section II-D.

A. Problem Statement

Our goal is to give an accurate prediction of traffic rule violations of other traffic participants, which we express as a multi-label classification problem [26, Sec. 8.1]. Traffic rule violations are predicted for a time horizon of H time steps. Let $w_f[k-L, k]$ be the input used for classification, i.e., the feature vectors $w_f[i] \in \mathbb{R}^F$ of the last L time steps. We define a classifier as $\phi : \mathbb{R}^{F \times L} \to \{0, 1\}^H$. An optimal classifier $\phi^*_\varphi(w_f[k-L, k]) = b^*$ for rule φ returns a classification result b^*, for which $\forall i \in \{1, ..., H\} : b^*_i = 0 \iff w_\mathcal{I}, k+i \models \varphi$ holds.

B. Overview

Using neural networks to predict traffic rule violations requires encoding the relevant environmental information as inputs. For some rules, e.g., speed limits, providing the vehicle state or a sequence of vehicle states and the speed limit may be sufficient. However, more complex rules require additional information, as we show in our evaluation in Section IV-B. End-to-end learning techniques [27] implicitly learn to extract relevant information from camera images. However, this increases training times and is computationally expensive, as many trivial tasks would first have to be learned, such as detecting traffic signs.

A traffic rule formulated in temporal logic exactly defines which quantities are relevant for its satisfaction and can be precisely evaluated. Rather than only evaluating whether a rule is satisfied, we compute the robustness degree of the corresponding STL formula to obtain information about whether one gets closer to violating a rule. We derive features $f_\varphi(w_\mathcal{I})[k] = \rho(\varphi, w_\mathcal{I}, k)$ from the robustness of $w_\mathcal{I}$ with respect to an STL formula φ as input for our neural network.

C. Robustness of Traffic Rule Predicates

The definition of the robustness degree for each predicate is necessary to calculate the robustness degree of a trace with respect to an STL formula. This section defines these values for the predicates used in the traffic rules G1, G2* and G3.

1) Threshold Predicates: Many predicates used in traffic rules are defined as inequality constraint $g(x) \geq 0$. Their robustness degree trivially follows from the equivalences defined in Section II-B: $\rho(g(x) \geq 0) = g(x)$. Table II shows robustness degrees that we examine for threshold predicates.

2) In-Same-Lane Predicate: If the vehicles with states x_p and x_q occupy at least one common lane, the predicate in-same-lane(x_p, x_q) is true. We first define the required lateral displacement of a vehicle with state x_i to be inside or outside the lanes occupied by another vehicle with state x_j, using the signed version of the directed Hausdorff distance

TABLE II: Robustness degree of threshold predicates used by traffic rules G1 to G3.

Predicate $\psi(x_p, [x_q])$	Robustness $\rho(\psi(x_p, [x_q]))$
keeps-fov-speed-limit(x_p)	$v_{\text{fov}} - \text{proj}_v(x_p)$
keeps-type-speed-limit(x_p)	$v_{\text{type}} - \text{proj}_v(x_p)$
keeps-lane-speed-limit(x_p)	$v_{\text{sl}}^{\max} - \text{proj}_v(x_p)$
keeps-brake-speed-limit(x_p)	$v_{\text{br}} - \text{proj}_v(x_p)$
brakes-abruptly(x_p)	$\text{proj}_a(x_p) - a_{\text{abrupt}}$
brakes-abruptly-relative(x_p, x_q)	$\text{proj}_a(x_q) - \text{proj}_a(x_p) + a_{\text{abrupt}}$
keeps-safe-distance-prec(x_p, x_q)	$\text{rear}(x_q) - \text{front}(x_p) - d_{\text{safe}}(x_p, x_q)$
in-front-of(x_p, x_q)	$\text{rear}(x_q) - \text{front}(x_p)$

to the lane boundary as follows:

$$\begin{aligned}&\text{distance-to-lanes}(x_i, x_j) = \\ &\quad \min\{\mathrm{d}_{\breve{H},\mathrm{l}}(\mathcal{O}(x_i), \mathrm{l}_{\mathrm{lb}}(\text{lanes}(x_j))), \\ &\qquad \mathrm{d}_{\breve{H},\mathrm{r}}(\mathcal{O}(x_i), \mathrm{l}_{\mathrm{rb}}(\text{lanes}(x_j)))\}.\end{aligned}$$

The robustness of the predicate is calculated as the minimum of the signed distances to the lanes occupied by the respective other vehicle:

$$\begin{aligned}&\rho(\text{in-same-lane}(x_p, x_q)) = \\ &\quad \min\{\text{distance-to-lanes}(x_p, x_q), \\ &\qquad \text{distance-to-lanes}(x_q, x_p)\}.\end{aligned} \tag{1}$$

Fig. 1a shows an example of the robustness for two vehicles.

3) Single-Lane Predicate: If the occupancy of the vehicle with state x_p only occupies a single lane, the predicate single-lane(x_p) is true. The robustness is similarly computed as for ρ(in-same-lane) by estimating the minimum required lateral displacement to satisfy the predicate. However, because all points of the vehicle occupancy must be contained in a single lane to satisfy the predicate, we use the minimum signed distance from a vehicle point to the reference lane as the robustness of the predicate:

$$\begin{aligned}&\rho(\text{single-lane}(x_p)) = \min\{ \\ &\quad \mathrm{d}_{\mathrm{l}}(\mathcal{O}(x_p), \mathrm{l}_{\mathrm{lb}}(\text{ref\_lane}(x_p))), \\ &\quad \mathrm{d}_{\mathrm{r}}(\mathcal{O}(x_p), \mathrm{l}_{\mathrm{rb}}(\text{ref\_lane}(x_p)))\}.\end{aligned} \tag{2}$$

Fig. 1b shows the robustness for the vehicle with index p in a single lane and p' occupying multiple lanes.

4) Cut-In Predicate: The predicate cut-in(x_p, x_q) defines whether the vehicle with state x_p enters the lane of the vehicle with state x_q. The robustness of this predicate can be directly derived from the equivalences of the logical operators in Section II-B and the previously defined functions ρ(single-lane) and ρ(in-same-lane):

$$\begin{aligned}&\rho(\text{cut-in}(x_p, x_q)) = \\ &\quad \min\big\{ -\rho(\text{single-lane}(x_p)), \rho(\text{in-same-lane}(x_p, x_q)), \\ &\qquad \max\{\min\{\text{proj}_d(x_q) - \text{proj}_d(x_p), \text{proj}_\theta(x_p)\}, \\ &\qquad\quad \min\{\text{proj}_d(x_p) - \text{proj}_d(x_q), -\text{proj}_\theta(x_p)\}\}\big\}.\end{aligned} \tag{3}$$

5) Precedes Predicate: For the modified version G2* of rule G2 we define a new predicate precedes(x_p, x_q), similarly as in [6]. It indicates that x_q is the direct predecessor of x_p in the same lane. We define a function returning the set of preceding vehicles:

$$\begin{aligned}&\text{predecessors}(x_p) = \\ &\quad \{x_i | i \in \mathcal{I}_{\neg p} \wedge \text{same-lane}(x_p, x_i) \wedge \text{in-front-of}(x_p, x_i)\}.\end{aligned}$$

Using the set of all predecessors, we define the predicate

$$\begin{aligned}&\text{precedes}(x_p, x_q) \iff \\ &\quad x_q = \underset{x_i \in \text{predecessors}(x_p)}{\arg\min} \rho(\text{in-front-of}(x_p, x_i)).\end{aligned} \tag{4}$$

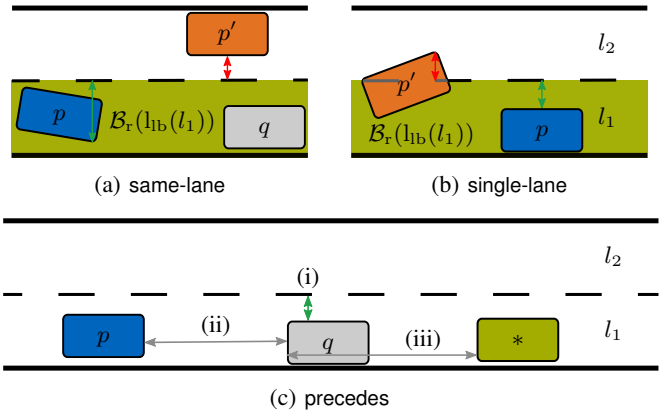
To determine the robustness of the predicate, we analyze the following necessary conditions: (i) being in the same lane and (ii) behind the vehicle with state x_q as well as (iii) vehicle with state x_q being the closest predecessor. The robustness of (i) and (ii) are obtained from (1) and Table II respectively, while (iii) is determined by finding the closest predecessor of the vehicle with state x_p with exception to the vehicle with state x_q:

$$x^* = \underset{x_i \in \text{predecessors}(x_p) \setminus \{x_q\}}{\arg\min} \rho(\text{in-front-of}(x_p, x_i)),$$

and calculating the longitudinal distance of the rearmost points of the two vehicles. The robustness is then defined as

$$\begin{aligned}\rho(\text{precedes}(x_p, x_q)) = \min\big\{&\rho(\text{in-same-lane}(x_p, x_q)), \\ &\rho(\text{in-front-of}(x_p, x_q)), \\ &\text{rear}(x^*) - \text{rear}(x_q)\big\}.\end{aligned} \tag{5}$$

Fig. 1c shows the respective distances used as the robustness of precedes(x_p, x_q). The minimal distance is shown with a green arrow, while the longer distances are shown with gray arrows.



(a) same-lane (b) single-lane

(c) precedes

Fig. 1: Visualization of distances used for the robustness of predicates. Green and red arrows indicate positive and negative robustness respectively.

D. Selection of the Target Vehicle

Some traffic rules are defined pairwise with respect to either all or at least one other traffic participant, e.g., *"keep a safe distance to all other vehicles"*, formalized by $\forall$ and $\exists$ quantification:

$$\begin{aligned}\varphi_\forall(x_p) &= \forall q \in \mathcal{I}_{\neg p} : \varphi(x_p, x_q) \\ \varphi_\exists(x_p) &= \exists q \in \mathcal{I}_{\neg p} : \varphi(x_p, x_q).\end{aligned}$$

As we not only use the robustness of the rule as features, but also the robustness of the predicates, this leads to a variable input size for our neural network stemming from the pairwise defined predicates. However, $\varphi_\forall(x_p)$ can be rewritten as a conjunction of the rule with respect to every other vehicle and $\varphi_\exists(x_p)$ as a disjunction, which results in the following robustness degrees:

$$\rho(\varphi_\forall, w_\mathcal{I}, k) = \min_{q \in \mathcal{I}_{\neg p}} \rho(\varphi(x_p, x_q), w_\mathcal{I}, k)$$
$$\rho(\varphi_\exists, w_\mathcal{I}, k) = \max_{q \in \mathcal{I}_{\neg p}} \rho(\varphi(x_p, x_q), w_\mathcal{I}, k).$$

This is equivalent to monitoring the most or least violating vehicle for each rule and time step:

$$\mathrm{tv}(\varphi(x_p), w_\mathcal{I}, k) = \begin{cases} \arg\min_{q \in \mathcal{I}_{\neg p}} \rho(\varphi(x_p, x_q), w_\mathcal{I}, k), & \text{if } \varphi = \varphi_\forall \\ \arg\max_{q \in \mathcal{I}_{\neg p}} \rho(\varphi(x_p, x_q), w_\mathcal{I}, k), & \text{if } \varphi = \varphi_\exists, \end{cases} \tag{6}$$

which we refer to as the target vehicle. In the next section, we present the evaluation of our approach.

IV. Numerical Experiments

To evaluate our approach, we predict traffic rule violations in highway traffic of the highD dataset [28]. It contains $110\,000$ extracted trajectories, recorded on six highway sections in Germany. Our simulation environment is CommonRoad [24], and we use RTAMT [29], to evaluate the robustness of STL formulas, and Tensorflow Keras[1] to implement neural networks. The proposed approach is compared to a baseline model that first predicts the trajectories of the vehicles using an RNN-based approach for trajectory prediction and afterward evaluates the predicted trajectories for compliance with the STL rules.

A. Machine Learning

1) Neural Networks: In our evaluation, we use two different types of neural networks: A multi-layer perceptron (MLP) with three hidden layers of sizes 462, 344, and 200, followed by an output layer of size H with one output neuron for each time step. The hidden layers use ReLu as an activation function, whereas the output layer uses the sigmoid activation function. Our second neural network is an RNN consisting of two layers with 373 and 110 gated recurrent units to capture patterns in the time domain, followed by three fully-connected layers with sizes 90, 24, 113, and an output layer with the same configuration as in the MLP.

Learning to predict rule violations from recorded data introduces the challenge of inherent class imbalance between violating and non-violating behavior with varying extent. Data-level and algorithm-level techniques exist to mitigate the problems arising from imbalanced data [30]. We use random under-sampling on the data level to increase the percentage of violating examples to $40\,\%$ by selecting all violating examples and randomly sampling from the non-violating partition. On the algorithm level, we use *focal loss* [31] with variables $\alpha = 0.25$ and $\gamma = 2$. This loss function was introduced to improve the performance of supervised learning on imbalanced datasets by focusing on examples that are more difficult to learn.

[1] https://www.tensorflow.org/

TABLE III: State normalization ranges and constants.

Scale	Value range	Unit	Constant	Value	Unit
Longitudinal distance	$[-200, 200]$	m	v_fov	50	m/s
Lateral distance	$[-20, 20]$	m	v_br	50	m/s
Velocity	$[-69.44, 69.44]$	m/s	v_type (truck)	22.22	m/s
Acceleration	$[-10.5, 10.5]$	m/s^2	a_abrupt	-2	m/s^2
Orientation	$[-\pi, \pi]$	rad	t_c	3	s

The neural networks are trained on an NVIDIA V100 GPU with 16 GB of memory. We process the training dataset with a batch size of 128 with the Adam optimizer [32], and a learning rate of 6×10^{-4}. The training is stopped after 20 epochs or once the area under the precision-recall curve (PR-AUC) [33, Sec. 3.3] on the validation partition has not improved by at least 1×10^{-4} for three epochs.

2) Features: We compare different features $f(w_\mathcal{I})$ as inputs for the neural network:

- $f_{\mathcal{P}_\varphi}$: the robustness degrees of the predicates used in φ;
- f_φ: the robustness degree of φ;
- $f_p = x_p - x_p[k]$: the difference between the past states and the current state of the predicted vehicle;
- $f_\mathrm{tv} = x_\mathrm{tv} - x_p$: the difference between the states of the target vehicle and the predicted vehicle.

We also use combinations of features, e.g., $\begin{pmatrix} f_{\mathcal{P}_\varphi} & f_\varphi \end{pmatrix}^\mathsf{T}$ by concatenating the feature components.

When using the quantitative semantics of STL, robustness degrees of predicates with different units and scales are directly compared, e.g., in (3): $\min\{\mathrm{proj}_d(x_q) - \mathrm{proj}_d(x_p), \mathrm{proj}_\theta(x_p)\}$, the minimum of a distance and an orientation. However, useful bounds for each unit and scale can be usually determined from the dynamic model or sensor ranges. Using these bounds, we normalize and clip terms of the form $\rho(g(x) \geq 0)$ according to their respective scale to obtain robustness values within the interval $[-1, 1]$. The bounds and constants for calculating the normalized values of $f_{\mathcal{P}_\varphi}$ and f_φ are shown in Table III. To improve the stability of the training, we additionally standardize all features.

3) Baseline Model: The prediction component of the baseline model is based on the highway trajectory prediction proposed in [34]. It uses the history of the past 25 time steps (5 s) of the state and the vehicle type of the predicted vehicle, as well as the difference of states, the vehicle type and time-to-collision of up to 9 surrounding vehicles as inputs. The neural network predicts the lateral position and longitudinal velocity within the prediction horizon, from which the state of the vehicle is reconstructed. The prediction network consists of a long short-term memory layer and two fully-connected time-distributed layers; for further details see [34]. After training, the average displacement error in the testing partition ranges from $0.01\,\mathrm{m}$ in the first time step to

$0.76\,\mathrm{m}$ in the last. We extend the prediction component with an STL monitor to obtain predictions for the satisfaction of the traffic rules. The combined model is referred to as the baseline model.

B. Evaluation

1) Data Preparation: We use the CommonRoad-highD converter[2] to create scenarios with a maximum length of $100\,\mathrm{s}$ and a time resolution of $0.2\,\mathrm{s}$. The resulting 1264 scenarios are randomly assigned to training, validation, and testing partitions in a proportion of 3:1:1. For every vehicle and time step of the dataset, we evaluate the robustness degree $\rho(\varphi, w_{\mathcal{I}}, k)$ of the STL formula $\varphi \in \{\mathrm{G1}, \mathrm{G2}^*, \mathrm{G3}\}$. In $11.47\,\%$ of all time steps, the safe distance to the preceding vehicle (G1) is violated, in $0.02\,\%$ vehicles brake unnecessarily ($\mathrm{G2}^*$) and in $36.65\,\%$ the speed limit is exceeded (G3). To create training instances with a fixed size from trajectories of variable length, we use a sliding window of length $L+H$ of which the first $L = 8$ time steps ($1.6\,\mathrm{s}$) are taken as the input of the prediction model, and the remaining $H = 20$ time steps ($4\,\mathrm{s}$) are the expected output used for supervised learning.

2) Results: We compare the ability to separate future traffic rule violations from conforming behavior of different types of neural networks and features, using the PR-AUC over the prediction horizon. The results of the comparison are shown in Fig. 2. Note, that for an uninformed classifier, which classifies each class with equal probability, the expected PR-AUC is equal to the fraction of the positive (violating) class [33, Sec. 3.3].

The highest performance is achieved for rule G3, which does not change significantly over the prediction horizon. This meets our expectation, as in a highway setting, velocities are mostly near-constant, violations are foreseeable, and occur for prolonged periods. For G3 and G1, both neural network types perform similarly well, but for G1, the prediction performance degrades with the prediction horizon.

The prediction of violations for rule $\mathrm{G2}^*$, exhibits significantly worse results. Two main reasons can explain this: The training set for rule $\mathrm{G2}^*$ has a high class imbalance, which imposes a challenge for the training of neural networks [30]. Furthermore, unnecessary braking is inherently difficult to predict as, by definition, there does not exist an apparent reason for braking. Thus, a characteristic pattern to be learned by the neural network might be absent. While the first half of the prediction horizon can still be partly predicted, our neural networks fail to recognize violations of rule $\mathrm{G2}^*$ for time steps more than $\approx 2\,\mathrm{s}$ in the future. However, the RNN has a minor advantage over the MLP during the first three time steps.

On average, we observed worse results for all rules, when only the state of the predicted and the target vehicle is used. This could be due to the absence of critical information, such as the local speed limit. Besides features f_p and f_{tv}, no strong domination of any other combination of features can be identified. Hence, the optimal combination is assumed to depend on the rule to be predicted. However, we find that $f_{\mathcal{P}_\varphi}$ is always included in the configuration with the highest average PR-AUC.

To compare our approach with the baseline model, we select the best combination of features for each neural network type based on the average PR-AUC. As missing a violation can be considered more critical than a false positive, we use the F2-score [33, Eq. 3.19], i.e., the weighted harmonic mean of recall and precision, where recall is considered twice as important as precision. We determine an optimal decision threshold based on the validation partition for the predicted probability of each time step within the prediction horizon for our neural networks.

Fig. 3 shows the result of the comparison. We observe the superior performance of the MLP and RNN on rule G1 and equal performance for G3, compared to the baseline model. For $\mathrm{G2}^*$, our models perform better as the baseline model until $2.4\,\mathrm{s}$ into the future, but detect fewer violations afterward. The two types of neural networks perform equally well for rule G1 and G3, but the MLP is slightly outperformed by the RNN when predicting violations of rule $\mathrm{G2}^*$.

V. Conclusion

This paper presented an approach for explicitly predicting violations of complex real-world traffic rules expressed in STL. The robustness degree of the STL formula and its predicates was used to obtain a generic and compact representation of features used as inputs for neural network prediction. By providing definitions for the robustness degree of the used predicates, we enhanced an existing formalization of three highway traffic rules for the evaluation with STL quantitative semantics. The robustness degree was also used to select a target vehicle when predicting pairwise defined traffic rules. We evaluated our approach on a dataset of highway driving using a multi-layer perceptron and recurrent neural network to predict traffic rule violations within a time horizon of $4\,\mathrm{s}$. Compared to only providing vehicle states or monitoring predicted vehicle trajectories, the results show that adding features from the historical robustness of predicates and rules improves the prediction.

Acknowledgements

The authors gratefully acknowledge the financial support by the German Research Foundation (DFG) grant AL 1185/7-1 and thank the Leibniz Supercomputing Centre (LRZ) of the Bavarian Academy of Sciences and Humanities (BAdW) for the provisioning and support of cloud computing infrastructure essential to this publication.

References

[1] B. Vanholme, D. Gruyer, B. Lusetti, *et al.*, "Highly Automated Driving on Highways Based on Legal Safety," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 1, pp. 333–347, 2013.

[2] M. Koschi and M. Althoff, "Set-Based Prediction of Traffic Participants Considering Occlusions and Traffic Rules," *IEEE Trans. Intell. Veh.*, vol. 6, no. 2, pp. 249–265, 2021.

[3] C. Pek, S. Manzinger, M. Koschi, *et al.*, "Using Online Verification to Prevent Autonomous Vehicles from Causing Accidents," *Nat. Mach. Intell.*, vol. 2, no. 9, pp. 518–528, 2020.

[2] `https://gitlab.lrz.de/tum-cps/dataset-converters`

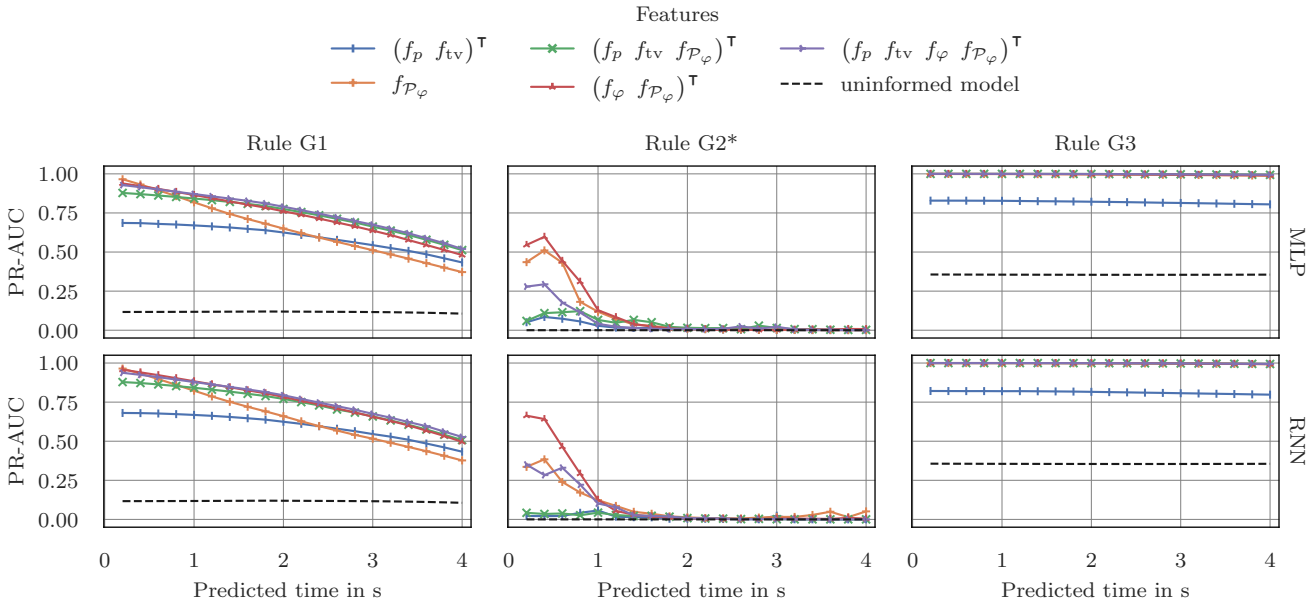


Fig. 2: Comparison of PR-AUC for different types of neural networks, input sets, rules and time steps.

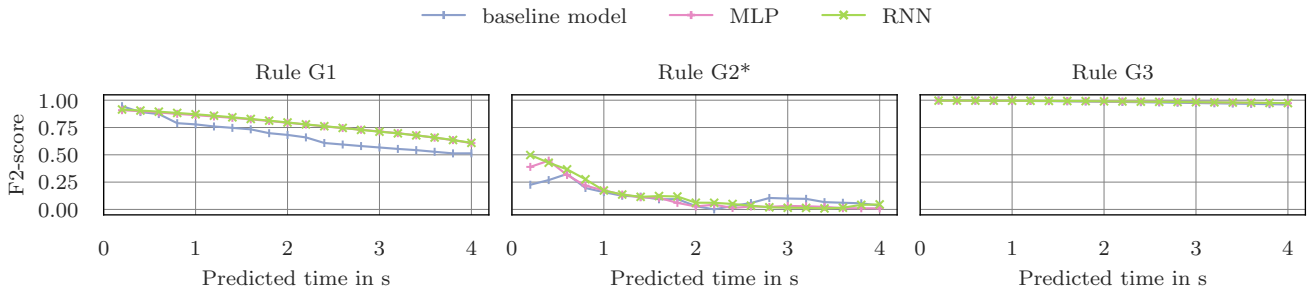


Fig. 3: Comparison of the best configurations of each neural network with the baseline model based on F2-score.

[4] S. Lefèvre, D. Vasquez, and C. Laugier, "A Survey on Motion Prediction and Risk Assessment for Intelligent Vehicles," *ROBOMECH J.*, vol. 1, no. 1, article no. 1, 2014.

[5] S. Mozaffari, O. Y. Al-Jarrah, M. Dianati, *et al.*, "Deep Learning-Based Vehicle Behavior Prediction for Autonomous Driving Applications: A Review," *IEEE Trans. Intell. Transp. Syst.*, to be published.

[6] K. Esterle, L. Gressenbuch, and A. Knoll, "Formalizing Traffic Rules for Machine Interpretability," in *Proc. IEEE Connected and Autom. Veh. Symp.*, 2020.

[7] S. Maierhofer, A.-K. Rettinger, E. C. Mayer, *et al.*, "Formalization of Interstate Traffic Rules in Temporal Logic," in *Proc. IEEE Intell. Veh. Symp.*, 2020, pp. 752–759.

[8] A. Rizaldi and M. Althoff, "Formalising Traffic Rules for Accountability of Autonomous Vehicles," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, 2015, pp. 1658–1665.

[9] A. Pnueli, "The Temporal Logic of Programs," in *Annu. Symp. Found. Comput. Sci.*, 1977, pp. 46–57.

[10] E. Plaku and S. Karaman, "Motion Planning with Temporal-logic Specifications: Progress and Challenges," *AI Commun.*, vol. 29, no. 1, pp. 151–162, 2016.

[11] R. Koymans, "Specifying Real-time Properties with Metric Temporal Logic," *Real-Time Syst*, vol. 2, no. 4, pp. 255–299, 1990.

[12] O. Maler and D. Ničković, "Monitoring Temporal Properties of Continuous Signals," in *Formal Techniques, Modelling and Anal. Timed Fault-Tolerant Syst.*, 2004, pp. 152–166.

[13] G. Fainekos and G. J. Pappas, "Robustness of Temporal Logic Specifications for Continuous-time Signals," *Theor. Comput. Sci.*, vol. 410, no. 42, pp. 4262–4291, 2009.

[14] M. Hekmatnejad, S. Yaghoubi, A. Dokhanchi, *et al.*, "Encoding and Monitoring Responsibility Sensitive Safety Rules for Automated Vehicles in Signal Temporal Logic," in *Proc. ACM-IEEE Int. Conf. Formal Methods and Models Syst. Design*, 2019, article no. 6.

[15] S. Shalev-Shwartz, S. Shammah, and A. Shashua, "On a Formal Model of Safe and Scalable Self-driving Cars," 2018. arXiv: 1708.06374 [cs, stat].

[16] K. Cho, T. Ha, G. Lee, *et al.*, "Deep Predictive Autonomous Driving Using Multi-Agent Joint Trajectory Prediction and Traffic Rules," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, 2019, pp. 2076–2081.

[17] X. Li, G. Rosman, I. Gilitschenski, *et al.*, "Vehicle Trajectory Prediction Using Generative Adversarial Network With Temporal Logic Syntax Tree Features," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 3459–3466, 2021.

[18] X. Qin and J. V. Deshmukh, "Clairvoyant Monitoring for Signal Temporal Logic," in *Formal Modeling and Anal. Timed Syst.*, 2020, pp. 178–195.

[19] Z. Yan, "Traj-ARIMA: A Spatial-time Series Model for Network-constrained Trajectory," in *Proc. Int. Workshop Comput. Transp. Sci.*, 2010, pp. 11–16.

[20] M. Ma, J. Stankovic, E. Bartocci, *et al.*, "CityPM: Predictive Monitoring with Logic-Calibrated Uncertainty for Smart Cities," 2020. arXiv: 2011.00384 [cs, eess].

[21] E. Bartocci, J. Deshmukh, A. Donzé, *et al.*, "Specification-Based Monitoring of Cyber-Physical Systems: A Survey on Theory, Tools and Applications," in *Lectures on Runtime Verification: Introductory and Advanced Topics*, Springer, 2018, pp. 135–175.
[22] O. Maler, D. Nickovic, and A. Pnueli, "Checking Temporal Properties of Discrete, Timed and Continuous Behaviors," in *Pillars of Computer Science*, Springer, 2008, pp. 475–505.
[23] E. Héry, S. Masi, P. Xu, *et al.*, "Map-based Curvilinear Coordinates for Autonomous Vehicles," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, 2017.
[24] M. Althoff, M. Koschi, and S. Manzinger, "CommonRoad: Composable Benchmarks for Motion Planning on Roads," in *Proc. IEEE Intell. Veh. Symp.*, 2017, pp. 719–726.
[25] A. Rizaldi, J. Keinholz, M. Huber, *et al.*, "Formalising and Monitoring Traffic Rules for Autonomous Vehicles in Isabelle/HOL," in *Proc. Int. Conf. Integrated Formal Methods*, 2017, pp. 50–66.
[26] M. Mohri, A. Rostamizadeh, and A. Talwalkar, "Multi-Class Classification," in *Foundations of Machine Learning*, MIT Press, 2012, pp. 183–208.
[27] M. Bojarski, D. Del Testa, D. Dworakowski, *et al.*, "End to End Learning for Self-Driving Cars," 2016. arXiv: 1604.07316 [cs].
[28] R. Krajewski, J. Bock, L. Kloeker, *et al.*, "The highD Dataset: A Drone Dataset of Naturalistic Vehicle Trajectories on German Highways for Validation of Highly Automated Driving Systems," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, 2018, pp. 2118–2125.
[29] D. Ničković and T. Yamaguchi, "RTAMT: Online Robustness Monitors from STL," in *Autom. Technol. Verification and Anal.*, 2020, pp. 564–571.
[30] J. M. Johnson and T. M. Khoshgoftaar, "Survey on Deep Learning with Class Imbalance," *J. Big Data*, vol. 6, no. 1, 2019.
[31] T.-Y. Lin, P. Goyal, R. Girshick, *et al.*, "Focal Loss for Dense Object Detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 318–327, 2020.
[32] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," 2017. arXiv: 1412.6980 [cs].
[33] A. Fernández, S. García, M. Galar, *et al.*, "Performance Measures," in *Learning from Imbalanced Data Sets*, Springer, 2018, pp. 47–61.
[34] F. Altché and A. d. L. Fortelle, "An LSTM Network for Highway Trajectory Prediction," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, 2017, pp. 353–359.
[35] Bundesministeriums der Justiz und für Verbraucherschutz, "Straßenverkehrsordnung (StVO)," 2013.
[36] Economic Commission for Europe Inland Transport Committee, "Convention on Road Traffic," 1968.

APPENDIX

For our work, we adopt the codification from [7] of the rules G1 and G3. However, we modify rule G2 as it is not robust towards accelerations of preceding vehicles. We argue that the rule should be modified such that braking with $\mathrm{proj}_a(x_p) > a_{\mathrm{abrupt}}$ (which is not considered braking abruptly) is always allowed and only braking stronger without justification (violation of safe distance or vehicle in front brakes abruptly) violates the rule of unnecessary braking. This is in accordance with the German road regulations (StVO) [35, §4(1)] and the Vienna Convention of Road Traffic (VCoRT) [36, §17(1)]. Additionally, we assume that this rule is only relevant towards the directly preceding vehicle, as otherwise breaking abruptly would be justified, even if a vehicle far in front would break abruptly. The modified version $\mathrm{G2}^*$ of rule G2 is shown in Table I.