Spatiotemporal Scene-Graph Embedding for Autonomous Vehicle Collision Prediction

Arnav Vaibhav Malawade, *Graduate Student Member, IEEE*, Shih-Yuan Yu, *Graduate Student Member, IEEE*, Brandon Hsu, Deepan Muthirayan, Pramod P. Khargonekar, *Life Fellow, IEEE*, and Mohammad Abdullah Al Faruque, *Senior Member, IEEE*

***Abstract*—In autonomous vehicles (AVs), early warning systems rely on collision prediction to ensure occupant safety. However, state-of-the-art methods using deep convolutional networks either fail at modeling collisions or are too expensive/slow, making them less suitable for deployment on AV edge hardware. To address these limitations, we propose SG2VEC, a spatiotemporal *scene-graph* embedding methodology that uses the graph neural network (GNN) and long short-term memory (LSTM) layers to predict future collisions via visual scene perception. We demonstrate that SG2VEC predicts collisions 8.11% more accurately and 39.07% earlier than the state-of-the-art method on synthesized data sets, and 29.47% more accurately on a challenging real-world collision data set. We also show that SG2VEC is better than the state of the art at transferring knowledge from synthetic data sets to real-world driving data sets. Finally, we demonstrate that SG2VEC performs inference 9.3× faster with an 88.0% smaller model, 32.4% less power, and 92.8% less energy than the state-of-the-art method on the industry-standard Nvidia DRIVE PX 2 platform, making it more suitable for implementation on the edge.**

***Index Terms*—ADAS, autonomous vehicles (AVs), collision prediction, graph learning, scene graph.**

I. Introduction

THE SYNERGY of artificial intelligence (AI) and the Internet of Things (IoT) has accelerated the advancement of autonomous vehicle (AV) technologies, which is expected to revolutionize transportation by reducing traffic and improving road safety [1], [2]. However, recent reports of AV crashes suggest that there are still significant limitations. For example, multiple fatal Tesla Autopilot crashes can primarily be attributed to perception system failures [3], [4]. Additionally, the infamous fatal collision between an Uber self-driving vehicle and a pedestrian can be attributed to perception and prediction failures by the AV [5]. These accidents (among others) have eroded public trust in AVs, and nearly, 50% or more of the public have expressed their mistrust in AVs [6]. Current statistics indicate that perception and prediction errors were factors in over 40% of driver-related crashes between conventional vehicles [7]. However, a significant number of reported AV collisions are also the result of these errors [8], [9]. Thus, in this article, we aim to improve the safety and acceptance of AVs by incorporating *scene-graphs* into the perception pipeline of collision prediction systems to improve scene understanding.

For the past several years, automotive manufacturers have begun equipping consumer vehicles with statistics-based collision avoidance systems based on calculated single behavior threat metrics (SBTMs), such as time to collision (TTC), time to react (TTR), etc. [10], [11]. However, these methods lack robustness since they make significant assumptions about the behavior of vehicles on the road. A very limiting assumption they make is that vehicles do not diverge from their current trajectories [11]. SBTMs can also fail in specific scenarios. For example, TTC can fail when following a vehicle at the same velocity within a very short distance [11]. As a result, these methods are less capable of generalizing and can perform poorly in complex road scenarios. Moreover, to reduce false positives, these systems are designed to respond at the last possible moment [12]. Under such circumstances, the AV control system can fail to take timely corrective actions [13] if the system fails to predict a collision or estimates the TTC inaccurately.

More effective collision prediction methods using deep learning (DL) have also been proposed in the literature. However, these approaches can be limited because they do not explicitly capture the relationships between the objects in a traffic scene. Understanding these relationships could be critical as it is suggested that a human's ability to understand complex scenarios and identify potential risks relies on cognitive mechanisms for representing structure and reasoning about interobject relationships [14]. These models also require large data sets that are often costly or unsafe to generate. Synthetic data sets are typically used to augment the limited real-world data to train the models in such cases [15]. However, these trained models must then be able to transfer the knowledge gained from synthetic data sets to real-world driving scenarios. Furthermore, DL models contain millions of parameters and require IoT edge devices with significant computational power and memory to run efficiently. Likewise, hosting these models on the cloud is infeasible because it requires persistent low-latency Internet connections.

Manuscript received June 2, 2021; revised July 27, 2021 and November 22, 2021; accepted January 3, 2022. Date of publication January 6, 2022; date of current version June 7, 2022. This work was supported in part by the National Science Foundation (NSF) under Award CMMI-1739503 and Award ECCS-1839429, and in part by the Graduate Assistance in Areas of National Need (GAANN) under Award P200A180052. *(Arnav Vaibhav Malawade and Shih-Yuan Yu contributed equally to this article.) (Corresponding author: Arnav Vaibhav Malawade.)*

The authors are with the Department of Electrical Engineering & Computer Science, University of California at Irvine, Irvine, CA 92697 USA (e-mail: malawada@uci.edu; shihyuay@uci.edu; bdhsu@uci.edu; dmuthira@uci.edu; pramod.khargonekar@uci.edu; alfaruqu@uci.edu).

Digital Object Identifier 10.1109/JIOT.2022.3141044

In summary, the key research challenges associated with AV collision prediction are as follows.

1) Capturing complex relationships between road participants.
2) Detecting future collisions early enough such that the AV can take corrective actions.
3) Generalizing to a wide range of traffic scenarios.
4) Developing algorithms that can run efficiently on automotive IoT edge devices.

In this work, we propose the use of *scene-graphs* to represent road scenes and model interobject relationships as we hypothesize that this will improve perception and scene understanding. Recently, several works have shown that graph-based methods that capture and model complex relationships between entities can improve the performance at high-level tasks, such as behavior classification [16], [17] and semantic segmentation [18]. *Scene-graphs* are used in many domains to abstract the locations and attributes of objects in a scene and model interobject relationships [16], [17], [19]. In our prior work [20], we demonstrated that a *scene-graph* sequence classification approach can better assess the subjective risk of driving clips compared to a conventional CNN+LSTM-based approach; our approach could also better generalize and transfer knowledge across data sets. This article extends the approach presented in [20] to collision prediction by enabling the prediction of future road states via changes to the temporal modeling components of the architecture and changes in the problem formulation. The *scene-graph* representation we propose represents traffic objects as nodes and the relationships between them as edges. The novelty of our *scene-graph* representation lies in our graph construction technique that is specifically designed for higher level AV decisions such as collision prediction. One of our key contributions is showing that *a graph-based intermediate representation can be very effective and efficient for higher level AV tasks directly related to driving decisions*.

Our proposed methodology for collision prediction, SG2VEC, is shown in Fig. 1. It combines the *scene-graph* representation with a graph-embedding architecture to generate a sequence of *scene-graph* embeddings for the sequence of visual inputs perceived by an AV. The graph embedding technique we use is based on the core multirelational graph convolutional network (MR-GCN) framework [16] adapted for the collision prediction problem. The sequence of graph embeddings is then input to a long short-term memory (LSTM) network to make the final prediction on the possibility of a future collision. To the best of our knowledge, our work is the first to propose using *scene-graphs* for early collision prediction.

Our article makes the following key research contributions.

1) We demonstrate that our SG2VEC collision prediction methodology significantly outperforms the current state of the art on simulated lane-change data sets and a very challenging real-world collision data set containing a wide range of driving actions, collision types, and weather/road conditions.
2) We demonstrate that SG2VEC can transfer knowledge gained from simulated data to real-world driving data more effectively than the state-of-the-art method.

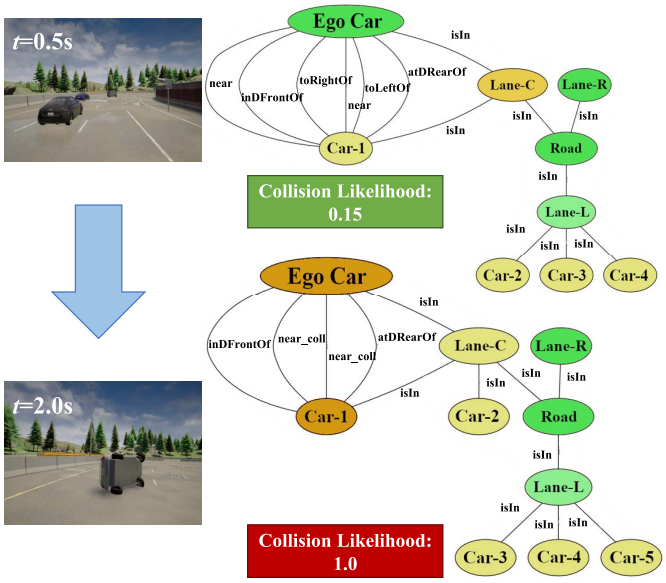

Fig. 1. How SG2VEC predicts collisions using *scene-graphs*. Each node's color indicates its attention score (importance to the collision likelihood) from orange (high) to green (low).

3) We show that SG2VEC performs faster inference and requires less power than the state-of-the-art method on the industry-standard Nvidia DRIVE PX 2 autonomous driving hardware platform, used in all 2016–2018 Tesla models for their Autopilot system [21].

II. Related Work

A. Early Collision Prediction

Since collision prediction is the key to the safety of AVs, a wide range of solutions has been proposed by academia and industry. As mentioned earlier, current consumer vehicles use statistics-based SBTMs for collision prediction but can perform poorly in complex situations [10], [11] or react too late to avoid collisions [12], [13]. Expanding on these approaches, companies, such as Mobileye and Nvidia, have proposed more comprehensive mathematical models for ensuring AV safety, namely, responsibility-sensitive safety (RSS) [22] and Nvidia Safety Force Field [23], respectively. However, these models are heavily rule based and can thus be fragile in complex situations with high uncertainty. Additionally, computing future trajectory constraints with RSS is nontrivial and can require vehicle-specific calibration [24].

Model-based probabilistic and DL approaches for collision prediction have also been proposed. For example, [25] proposes a model-based probabilistic technique that uses the roadway geometry, ego trajectory, and position/velocity of road objects to predict future object positions. However, this model is highly conservative and is likely to have a high false-positive rate. Similarly, [26] and [27] use model-based approaches but require significant domain knowledge about the driving scene, such as road geometry information as well as accurate vehicle position and velocity information. Wang *et al.* [28] proposed a DL collision prediction approach. Still, due to its use of preprocessed trajectory data captured from cameras

overlooking a highway, it is not ego-centric and cannot be practically used for on-vehicle collision prediction. In a different approach, [29] proposes a deep predictive model (DPM) that used a Bayesian convolutional LSTM (ConvLSTM) for collision risk assessment where image data, vehicle telemetry data, and driving inputs were all factors in the risk assessment decision. However, this approach was only evaluated on simulated street scenes containing two vehicles and no other dynamic objects. Thus, DPM's performance may suffer when evaluated on more complex road scenarios.

In contrast to these existing works, we propose SG2VEC, which captures structural and relational information of a road scene in a *scene-graph* representation and computes a spatiotemporal embedding to predict collisions. Additionally, we perform experiments that were not done in many prior works, such as evaluating each model's capability to transfer knowledge, efficiency on AV hardware, performance on a complex real-world crash data set, and ability to predict collisions early. We primarily compare our methodology with the DPM as it is the state-of-the-art data-driven collision prediction framework for AVs that considers both spatial and temporal factors. Although the DPM uses multiple modalities for sensing, the results in [29] show that it achieves an accuracy (of 81.95%) that is just 0.24% less using just the image sensing modality. In this work, we compare our proposed SG2VEC methodology and the DPM on image-only data sets, which is fair because the DPM's performance does not vary much with the inclusion of other modalities.

B. AV Scene-Graphs and Optimization Techniques

Several works have proposed graph-based methods for scene understanding. For example, [16] proposed an MR-GCN that uses both spatial and temporal information to classify vehicle driving behavior. Similarly, in [17], an *Ego-Thing* and *Ego-Stuff* graph are used to model and classify the ego vehicle's interactions with moving and stationary objects, respectively. In our prior work, we demonstrated that a *scene-graph* sequence embedding approach assesses driving risk better than the state-of-the-art CNN-LSTM approach [20]. In [20], we utilized an architecture consisting of MR-GCN layers for spatial modeling and an LSTM with attention for temporal modeling; however, this architecture was only capable of performing binary sequence-level classification over a complete video clip. Thus, although our prior architecture could accurately assess the subjective risk of complete driving sequences, it was not capable of predicting the future state of a scene.

Current autonomous driving systems consume a substantial amount of power (up to 500 W for the Nvidia DRIVE AGX Pegasus), demanding more robust cooling and power delivery mechanisms. Thus, many have tried to optimize AV tasks for efficiency without sacrificing performance. Existing approaches have proposed methods for jointly optimizing power consumption and latency for localization [30], perception [31], and control [32]. However, to the best of our knowledge, no work has explored this optimization for AV safety systems, such as collision prediction systems.

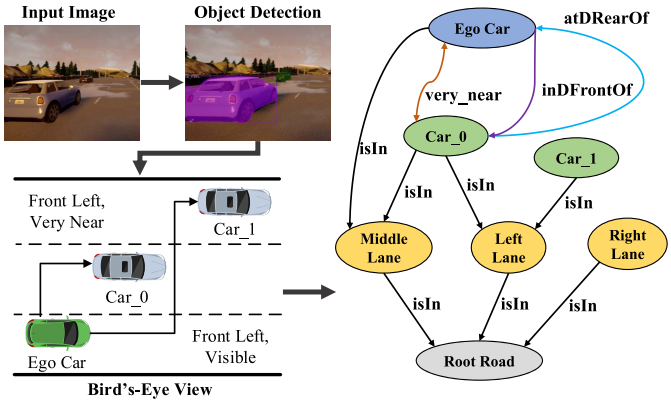

Fig. 2. Illustration of our *scene-graph* extraction process.

III. SCENE-GRAPH EMBEDDING METHODOLOGY

In SG2VEC, we formulate the problem of collision prediction as a time-series classification problem where the goal is to predict if a collision will occur in the *near future*. Our goal is to accurately model the spatiotemporal function f, where

$$\mathbf{Y_n} = f(\{I_1, \ldots, I_{n-1}, I_n\}), \mathbf{Y_n} \in \{0, 1\}, \text{for } n > 2 \qquad (1)$$

where $\mathbf{Y_n} = 1$ implies a collision in the near future and $\mathbf{Y_n} = 0$ otherwise. Here, the variable I_n denotes the image captured by the onboard camera at time n. The interval between each frame varies with the camera sampling rate.

SG2VEC consists of two parts (Fig. 3): 1) the *scene-graph* extraction and 2) collision prediction through spatiotemporal embedding, described in Sections III-A and III-B, respectively.

A. Scene-Graph Extraction

The first step of our methodology is the extraction of *scene-graphs* for the images of a driving scene. The extraction pipeline forms the *scene-graph* for an image as in [33] and [34] by first detecting the objects in the image and then identifying their relations based on their attributes. The difference from prior works lies in the construction of a *scene-graph* that is designed for higher level AV decisions. We propose extracting a *minimal* set of relations, such as directional relations and proximity relations. From our design space exploration, we found that adding many relation edges to the *scene-graph* adds noise and impacts convergence while using too few relation types reduces our model's expressivity. The best approach we found across applications involves constructing mostly ego-centric relations for a moderate range of relation types. Fig. 2 shows an example of the graph extraction process.

We denote the extracted *scene-graph* for the frame I_n by $G_n = \{O_n, A_n\}$. Each *scene-graph* G_n is a directed, heterogeneous multigraph, where O_n denotes the nodes and A_n is the adjacency matrix of the graph G_n. As shown in Fig. 2, nodes represent the identified objects, such as lanes, roads, traffic signs, vehicles, pedestrians, etc., in a traffic scene. The adjacency matrix A_n indicates the pairwise relations between each object in O_n. The extraction pipeline first identifies the objects O_n by using Mask R-CNN [35]. Then, it generates an

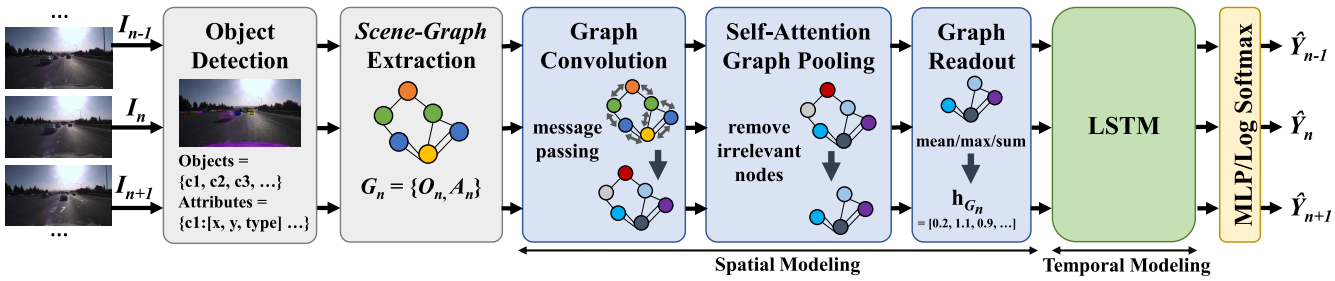

Fig. 3. Illustration of SG2VEC's architecture.

inverse perspective mapping (also known as a "birds-eye view (BEV)" projection) of the image to estimate the locations of objects relative to the ego car, which are used to construct the pairwise relations between objects in A_n. For each camera angle, we calibrate the BEV projection settings using known fixed distances, such as the lane length and width, as defined by the highway code. This enables us to estimate longitudinal and lateral distances accurately in the projection. For data sets captured by a single vehicle, this step only needs to be performed once. However, for data sets with a wide range of camera angles such as the *620-dash* data set introduced later in this article, this process needs to be performed once per vehicle. With a human operator, we found that this calibration step takes approximately 1 min per camera angle on average.

The extraction pipeline identifies three kinds of pairwise relations: 1) *proximity* relations (e.g., *visible*, *near*, *very_near*, etc.); 2) *directional* (e.g., *Front_Left*, *Rear_Right*, etc.) relations; and 3) *belonging* (e.g., car_1 *isIn* left_lane) relations. Two objects are assigned the *proximity* relation, $r \in$ {*Near_Collision* (4 ft.), *Super_Near* (7 ft.), *Very_Near* (10 ft.), *Near* (16 ft.), *Visible* (25 ft.)} provided the objects are physically separated by a distance that is within that relation's threshold. The *directional relation*, $r \in$ {*Front_Left*, *Left_Front*, *Left_Rear*, *Rear_Left*, *Rear_Right*, *Right_Rear*, *Right_Front*, *Front_Right*}, is assigned to a pair of objects, in this case between the ego-car and another car in the view, based on their relative orientation and only if they are within the *Near* threshold distance from one another. Additionally, the *isIn* relation identifies which vehicles are on which lanes (see Fig. 2). We use each vehicle's horizontal displacement relative to the ego vehicle to assign vehicles to either the *Left Lane*, *Middle Lane*, or *Right Lane* using the known lane width. Our abstraction only considers three-lane areas, and, as such, we map vehicles in all left lanes and all right lanes to the same *Left Lane* node *Right Lane* node, respectively. If a vehicle overlaps two lanes (i.e., during a lane change), it is mapped to both lanes.

B. Collision Prediction

As shown in Fig. 3, in our collision prediction methodology, each image I_n is first converted into a *scene-graph* $G_n = \{O_n, A_n\}$ with the pipeline mentioned in Section III-A. Each node $v \in O_n$ is initialized by a one-hot vector (*embedding*), denoted by $\mathbf{h}_v^{(0)}$. Then, the MR-GCN [36] layers are used to update these embeddings via the edges in A_n. Specifically, the lth MR-GCN layer computes the node embedding for each node v, denoted as $\mathbf{h}_v^{(l)}$, as follows:

$$\mathbf{h}_v^{(l)} = \Phi_0 \cdot \mathbf{h}_v^{(l-1)} + \sum_{r \in \mathbf{A_n}} \sum_{u \in \mathbf{N}_r(v)} \frac{1}{|\mathbf{N}_r(v)|} \Phi_r \cdot \mathbf{h}_u^{(l-1)} \tag{2}$$

where $N_r(v)$ denotes the set of neighbors of node v with respect to the relation $r \in A_n$, Φ_r is a trainable relation-specific transformation for relation r, and Φ_0 is the self-connection for each node v that accounts for the influence of $\mathbf{h}_v^{(l-1)}$ on $\mathbf{h}_v^{(l)}$ [36]. After the input is passed through multiple MR-GCN layers, the set of node embeddings output by each layer is collected and concatenated along the feature dimension to produce the final embedding for each node v, denoted by $\mathbf{H}_v^L = \mathbf{CONCAT}(\{\mathbf{h}_v^{(l)}\} | l = 0, 1, \ldots, L)$, where L is the index of the last layer. Thus, if the model uses two MR-GCN layers with an output size of 64, the final embedding for each node will contain 128 features.

The final embeddings for *scene-graph* G_n, denoted by $\mathbf{X}_n^{\text{prop}}$, are then passed through a graph pooling layer to filter out irrelevant nodes from the graph, creating the pooled set of node embeddings $\mathbf{X}_n^{\text{pool}}$ and their edges $\mathbf{A}_n^{\text{pool}}$. The pooling layer is implemented as follows:

$$\alpha = \mathbf{SCORE}\left(\mathbf{X}_n^{\text{prop}}, \mathbf{A}_n^{\text{prop}}\right) \tag{3}$$

$$\mathbf{P} = \text{top}_k(\alpha) \tag{4}$$

where **SCORE** can either be implemented as a top-k pooling (*Top-K*) [37] or self-attention graph pooling function (*SAGPool*) [38], α contains the score of each node in G_n, and $\mathbf{P}$ is the set of k highest scoring nodes in G_n. After pooling, the node embeddings and adjacency matrix are denoted as $\mathbf{X}_t^{\text{pool}}$ and $\mathbf{A}_\mathbf{t}^{\text{pool}}$ computed as follows:

$$\mathbf{X}_t^{\text{pool}} = \left(\mathbf{X}_t^{\text{prop}} \odot \tanh(\alpha)\right)_\mathbf{P} \tag{5}$$

$$\mathbf{A}_\mathbf{t}^{\text{pool}} = \mathbf{A}_\mathbf{t}^{\text{prop}}{}_{(\mathbf{P},\mathbf{P})} \tag{6}$$

where $\odot$ represents elementwise multiplication, $()_\mathbf{P}$ refers to the operation that selects only the subset of nodes defined by P, and $()_{(\mathbf{P},\mathbf{P})}$ refers to the formation of the adjacency matrix between the nodes in this subset.

Then, for each *scene-graph* G_n, the corresponding $\mathbf{X}_n^{\text{pool}}$ is passed through the graph **READOUT** operation that condenses the node embeddings to a single graph embedding $\mathbf{h}_{G_n}$ as follows:

$$\mathbf{h}_{G_n} = \mathbf{READOUT}(X_t^{\text{pool}}) \tag{7}$$

Fig. 4. Examples of driving scenes from our (a) synthetic data sets, (b) typical real-world data set, and (c) complex real-world data set. In (a), all driving scenes occur on highways with the same camera position and clearly defined road markings; lighting and weather are dynamically simulated in CARLA. In (b), driving scenes occur on multiple types of clearly marked roads but lighting, camera angle, and weather are consistent across scenes. (c) contains a much broader range of camera angles as well as more diverse weather and lighting conditions, including rain, snow, and night-time driving; it also contains a large number of clips on unpaved or unmarked roadways, as shown.

where **READOUT** can be an operation such as averaging (*mean-readout*), summation (*add-readout*), or retrieving the maximum (*max-readout*) in each feature dimension for the set of pooled node embeddings X_t^{pool}.

Then, this spatial embedding $\mathbf{h}_{G_n}$ is passed to the temporal model (LSTM) to generate a spatiotemporal embedding z_n as follows:

$$z_n, s_n = \mathbf{LSTM}\left(h_{G_n}, s_{n-1}\right) \tag{8}$$

where s_{n-1} represents the hidden state of the LSTM after the previous time step. For each timestamp n, the LSTM produces an output embedding z_n and updates its hidden state s_n. Since the hidden state is carried over to the next time step $n+1$ and used to compute z_{n+1}, it enables the LSTM to model how the spatial embeddings h_{G_n} change over time.

Finally, each spatiotemporal embedding z_n is then passed through a multilayer perceptron (MLP) that outputs each class's confidence value. The two outputs of the MLP are compared, and $\hat{Y}_n$ is set to the index of the class with the greater confidence value (0 for no-collision or 1 for collision). During training, we calculate the cross-entropy loss between each set of nonbinarized outputs $\hat{Y}_n$ and the corresponding labels for backpropagation.

IV. Experimental Results

This section provides extensive experimental results to demonstrate SG2VEC's performance, efficiency, and transferability compared to the state-of-the-art collision prediction model, DPM [29]. For SG2VEC, we used two MR-GCN layers, each of size 64, one *SAGPooling* layer with a pooling ratio of 0.25, one *add-readout* layer, one LSTM layer with hidden size 20, one MLP layer with an output of size 2, and a LogSoftmax to generate the final confidence value for each class. For the DPM, we followed the architecture used in [29], which uses one $64 \times 64 \times 5$ ConvLSTM layer, one $32 \times 32 \times 5$ ConvLSTM layer, one $16 \times 16 \times 5$ ConvLSTM layer, one MLP layer with an output size of 64, one MLP layer with an output size of 2, and a Softmax to generate the final confidence value. For both models, we used a dropout of 0.1 and ReLU activation. The learning rates were 0.00005 for SG2VEC and 0.0001 for DPM. We ran the experiments shown in Sections IV-B and IV-C on a Windows PC with an AMD Ryzen Threadripper 1950X processor, 16-GB RAM, and an Nvidia GeForce RTX 2080 Super GPU.

A. Data Set Preparation

We prepared three types of data sets for our experiments: 1) synthesized data sets; 2) a typical real-world driving data set; and 3) a complex real-world driving data set. Examples from each data set are shown in Fig. 4. Our synthetic data sets focus on the highway lane change scenario as it is a common AV task. To evaluate the transferability of each model from synthetic data sets to real-world driving, we prepared a typical real-world data set containing lane-change driving clips. Finally, we prepared the complex real-world driving data set to evaluate each model's performance on a challenging data set containing a broad spectrum of collision types, road conditions, and vehicle maneuvers. All data sets were collected at a 1280×720 resolution, and each clip spans 1–5 s.

1) Synthetic Data Sets: To synthesize the data sets, we developed a tool using CARLA [15], an open-source driving simulator, and CARLA Scenario Runner[1] to generate lane change video clips with/without collisions. We generated a wide range of simulated lane changes with different numbers of cars, pedestrians, weather and lighting conditions, etc. We also customized each vehicle's driving behavior, such as their intended speed, probability of ignoring traffic lights, or the chance of avoiding collisions with other vehicles. We generated two synthetic data sets: 1) a *271-syn* dataset and

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

2) a *1043-syn* data set, containing 271 and 1043 video clips, respectively. These data sets have no-collision:collision label distributions of 6.12:1 and 7.91:1, respectively. In addition, we subsampled the *1043-syn* data set to create *306-syn*: a balanced data set that has a 1:1 distribution. Our synthetic *scene-graph* data sets[2] and our source code[3] are open source and available online.

2) Typical Real-World Driving Data Set: This data set, denoted as *571-honda*, is a subset of the Honda driving data set (HDD) [39] containing 571 lane-change video clips from real-world driving with a distribution of 7.21:1. The HDD was recorded on the same vehicle during mostly safe driving in the California Bay Area.

3) Complex Real-World Driving Data Set: Our complex real-world driving data set, denoted as *620-dash*, contains very challenging real-world collision scenarios drawn from the detection of traffic anomaly data set [40]. This data set contains a wide range of drivers, car models, driving maneuvers, weather/road conditions, and collision types, as recorded by onboard dashboard cameras. Since the original data set contains only collision clips, we prepared *620-dash* by splitting each clip in the original data set into two parts: 1) the beginning of the clip until 1 s before the collision and 2) from 1 s before the collision until the end of the collision. We then labeled part 1) as "no-collision" and part 2) as "collision." The *620-dash* data set contains 315 collision video clips and 342 noncollision driving clips.

4) Labeling and Preprocessing: We labeled the synthetic data sets and the *571-honda* data set using human annotators. The final label assigned to a clip is the average of the labels assigned by the human annotators rounded to 0 (no collision) and 1 (collision/near collision). Each frame in a video clip is given a label identical to the entire clip's label to train the model to identify the preconditions of a future collision.

For SG2VEC, all the data sets were preprocessed using the *scene-graph* extraction pipeline mentioned in Section III-A to construct the *scene-graphs* for each video clip. For a given sequence, SG2VEC can leverage the full history of prior frames for each new prediction. For the DPM, the data sets were preprocessed to match the input format used in its original implementation [29]. Thus, the DPM uses 64×64 grayscale versions of the clips in the data sets turned into sets of subsequences J_n for a clip of length l defined as follows:

$$J_n = \{I_n, I_{n+1}, I_{n+2}, I_{n+3}, I_{n+4}\}, \text{for } n \in [1, l-4]. \quad (9)$$

Since DPM only uses five prior frames to make each prediction, we also present results for SG2VEC using the same length of history, denoted as SG2VEC (5-frames) in the results.

B. Collision Prediction Performance

We evaluated SG2VEC and the DPM using classification accuracy, area under the ROC curve (AUC) [41], and Matthews correlation coefficient (MCC) [42]. MCC is considered a balanced measure of performance for binary classification even on data sets with significant class imbalances. The MCC score outputs a value between −1.0 and 1.0, where 1.0 corresponds to a perfect classifier, 0.0 to a random classifier, and −1.0 to an always incorrect classifier. Although class reweighting helps compensate for the data set imbalance during training, classification accuracy is typically less reliable for imbalanced data sets, so the primary metric we use to compare the models is MCC. We used stratified fivefold cross-validation to produce the final results shown in Table I and Fig. 5.

TABLE I
CLASSIFICATION ACCURACY, AUC, AND MCC FOR SG2VEC (OURS) AND DPM

Dataset	Model	Accuracy	AUC	MCC
271-syn	SG2VEC (5-frames)	**0.8979**	**0.9541**	**0.5362**
271-syn	SG2VEC	0.8812	0.9457	0.5145
271-syn	DPM	0.8733	0.8939	0.2160
306-syn	SG2VEC (5-frames)	0.7946	0.8653	0.5790
306-syn	SG2VEC	**0.8372**	**0.9091**	**0.6812**
306-syn	DPM	0.6846	0.6881	0.3677
1043-syn	SG2VEC (5-frames)	**0.9142**	**0.9623**	0.5323
1043-syn	SG2VEC	0.9095	0.9477	**0.5385**
1043-syn	DPM	0.8834	0.9175	0.2912
620-dash	SG2VEC (5-frames)	0.6534	0.7113	0.3053
620-dash	SG2VEC	**0.7007**	**0.7857**	**0.4017**
620-dash	DPM	0.4890	0.4717	-0.0366

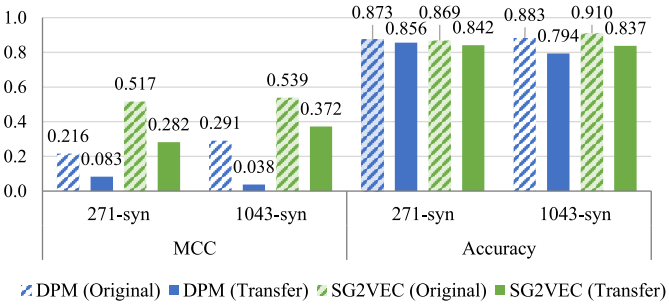

Fig. 5. Performance after transferring the models trained on synthetic *271-syn* and *1043-syn* data sets to the real-world *571-honda* data set.

1) Synthetic Data Sets: The performance of SG2VEC and the DPM on our synthetic data sets is shown in Table I. We find that our SG2VEC achieves higher accuracy, AUC, and MCC on every data set, even when only using five prior frames as input. In addition to predicting collisions more accurately, SG2VEC also infers 5.5x faster than the DPM on average. We attribute this to the differences in model complexity between our SG2VEC architecture and the much larger DPM model. Interestingly, SG2VEC (5-frames) achieves slightly better accuracy and AUC than SG2VEC on the imbalanced data sets and slightly lower overall performance on the balanced data sets. This is likely because the large number of safe lane changes in the imbalanced data sets adds noise during training and makes the full-history version of the model perform slightly worse. However, the full model can learn long-tail patterns for collision scenarios and perform better on the balanced data sets.

The DPM achieves relatively high accuracy and AUC on the imbalanced *271-syn* and *1043-syn* data sets, but suffers significantly on the balanced *306-syn* data set. This drop indicates that the DPM could not identify the minority class (collision) well and tended to overpredict the majority class (no collision). In terms of MCC, the DPM scores higher on the *306-syn* data

[2]https://dx.doi.org/10.21227/c0z9-1p30
[3]https://github.com/AICPS/sg-collision-prediction

set than what it scores on the other data sets. This result is because the *306-syn* data set has a balanced class distribution compared to the other data sets, which could enable the DPM to improve its prediction accuracy on the collision class.

In contrast, the SG2VEC methodology performs well on both balanced and imbalanced synthetic data sets with an average MCC of 0.5860, an average accuracy of 87.97%, and an average AUC of 0.9369. Since MCC is scaled from −1.0 to 1.0, SG2VEC achieves a 14.72% higher average MCC score than the DPM model.

The results from our SG2VEC ablation study are shown in Table II and support our hypothesis that both spatial modeling with MRGCN and temporal modeling with LSTM are core to the SG2VEC's collision prediction performance. However, the MRGCN appears to be slightly more critical to performance than the LSTM. Interestingly, the choice of pooling layer (no pooling, Top-*K* pooling, or SAG Pooling) does not seem to significantly affect performance at this task as long as LSTM is used; when no LSTM is used, SAG Pooling presents a clear performance improvement.

2) Complex Real-World Data Set: The performance of both the models significantly drops on the highly complex real-world *620-dash* data set due to the variations in the driving scenes and collision scenarios. This drop is to be expected as this data set contains a wide range of driving actions, road environments, and collision scenarios, increasing the difficulty of the problem significantly. We took several steps to try and address this performance drop. First, we improved the BEV calibration on this data set in comparison to the other data sets. Since the varying camera angles and road conditions in this data set impact our ability to properly calibrate SG2VEC's BEV projection in a single step, we created custom BEV calibrations for each clip in the data set, which improved the performance somewhat. However, as shown in Fig. 4(c), a significant part of the data set consists of driving clips on roads without any discernible lane markings, such as snowy, unpaved, or unmarked roadways. These factors make it challenging to correlate known fixed distances (i.e., the width and length of lane markings) with the projections of these clips. To further improve the performance on this particular data set, we performed extensive architecture and hyperparameter tuning. We found that with one MRGCN layer of size 64, one LSTM layer with hidden size 100, no SAGPooling layer, and a high learning rate and batch size, we achieved significantly better performance than the model architecture discussed in the beginning of Section IV (two MRGCN layers of size 64, one LSTM layer with hidden size 20, and a SAGPooling layer with a keep ratio of 0.5). We believe this indicates that the temporal features of each clip in this data set are more closely related to collision likelihood than the spatial features in each clip. As a result, the additional spatial modeling components were likely causing overfitting and skewing the spatial embedding output. The spatial embeddings remained more general with a simpler spatial model (one MRGCN and no SAGPooling). This change, combined with using a larger LSTM layer, enabled the model to capture more temporal features when modeling each clip and better generalize to the testing set. The model performance on this data set and similar data sets could likely be improved by acquiring more consistent data via higher resolution cameras with fixed camera angles and more accurate BEV projection approaches. However, as collisions are rare events, there are little to no data sets containing real-world collisions that meet these requirements. Despite these limitations, SG2VEC outperforms the DPM model by a significant margin, achieving 21.17% higher accuracy, 31.40% higher AUC, and a 21.92% higher MCC score. Since DPM achieves a negative MCC score, its performance on this data set is worse than that of a random classifier (MCC of 0.0). Consistent with the synthetic data set results, *sg2vec* using all frames performs better on the balanced *620-dash* data set than SG2VEC (5-frames). Overall, these results show that on very challenging and complex real-world driving scenarios, SG2VEC can perform much better than the current state of the art.

TABLE II
SG2VEC ABLATION STUDY ON THE *1043-syn* DATA SET

Experiment	Spatial Model	Graph Pooling	Temporal Model	Acc.	MCC
Ablation Study	MLP	none	none	0.7605	0.2612
	MLP	none	LSTM	0.7660	0.2874
	MRGCN	none	none	0.8605	0.4792
	MRGCN	none	LSTM	**0.8931**	**0.5561**
Graph Attn. and Pooling	MRGCN	Top-K	none	0.8288	0.3458
	MRGCN	SAGPool	none	0.8738	0.5032
	MRGCN	Top-K	LSTM	**0.9014**	**0.5565**
	MRGCN	SAGPool	LSTM	**0.9076**	**0.5407**

TABLE III
ATP FOR COLLISIONS

Dataset	Model	ATP	Avg. Seq. Len.	Ratio
271-syn	**SG2VEC (Ours)**	10.004	33.920	**0.2949**
271-syn	DPM	17.399	32.899	0.5289
1043-syn	**SG2VEC (Ours)**	6.442	37.343	**0.1725**
1043-syn	DPM	9.018	37.856	0.2382

3) Time of Prediction: Since collision prediction is a time-sensitive problem, we evaluated our methodology and the DPM on their average time-of-prediction (ATP) for video clips containing collisions. To calculate the ATP, we recorded the first frame index in each collision clip when the model correctly predicts that a collision would occur. We then averaged these indices and compared them with the average collision video clip length. Essentially, ATP gives an estimate of how early each model can predict a future collision. These results are shown in Table III. On the *1043-syn* data set, SG2VEC achieves 0.1725 for the ratio of the ATP and the average sequence length while the DPM achieves a ratio of 0.2382, indicating that SG2VEC predicts future collisions 39.07% earlier than the DPM on average. In the context of real-world collision prediction, the average sequence in the *1043-syn* data set represents 1.867 s of data. Thus, our methodology predicted collisions 122.7 ms earlier than DPM on average. This extra time can be critical for ensuring that the AV avoids an impending collision.

C. Transferability From Synthetic to Real-World Data Sets

The collision prediction models trained on simulated data sets must be transferable to real-world driving as it can differ

significantly from simulations. To evaluate each model's ability to transfer knowledge, we trained each model on a synthetic data set before testing it on the *571-honda* data set. No additional domain adaptation was performed. We did not evaluate transferability to the *620-dash* data set because it contains a wide range of highly dynamic driving maneuvers that were not present in our synthesized data sets. As such, evaluating transferability between our synthesized data sets and the 620-dash data set would yield poor performance and would not provide insight. Fig. 5 compares the accuracy and MCC for both the models on each training data set and the *571-honda* data set after transferring the trained model.

We observe that the SG2VEC model achieves a significantly higher MCC score than the DPM model after the transfer, suggesting that our methodology can better transfer knowledge from a synthetic to a real-world data set compared to the state-of-the-art DPM model. The drop in MCC values observed for both the models when transferred to the *571-honda* data set can be attributed to the characteristic differences between the simulated and real-world data sets; the *571-honda* data set contains a more heterogeneous set of road environments, lighting conditions, driving styles, etc., so a drop in performance after the transfer is expected. We also note that the MCC score for the SG2VEC model trained on *271-syn* data set drops more than the model trained on the *1043-syn* data set after the transfer, likely due to the smaller training data set size. Regarding accuracy, the SG2VEC model trained on *1043-syn* achieves 4.37% higher accuracy and the model trained on *271-syn* data set achieves 1.47% lower accuracy than the DPM model trained on the same data sets. The DPM's similar accuracy after transfer likely results from the class imbalance in the *571-honda* data set. Overall, we hypothesize that SG2VEC's use of an intermediate representation (i.e., *scene-graphs*) inherently improves its ability to generalize and thus, results in an improved ability to transfer knowledge compared to CNN-based DL approaches.

D. Evaluation on Industry-Standard AV Hardware

To demonstrate that the SG2VEC is implementable on industry-standard AV hardware, we measured its inference time (milliseconds), model size (kilobytes), power consumption (watts), and energy consumption per frame (milli-joules) on the industry-standard Nvidia DRIVE PX 2 platform, which was used by Tesla for their Autopilot system from 2016 to 2018 [21]. Our hardware setup is shown in Fig. 6. For the inference time, we evaluated the average inference time (AIT) in milliseconds taken by each algorithm to process each frame. We recorded power usage metrics using a power meter connected to the power supply of the PX 2. To ensure that the reported numbers only reflected each model's power consumption and not that of background processes, we subtracted the hardware's idle power consumption from the averages recorded during each test. For a fair comparison, we captured the metrics for the core algorithm (i.e., the SG2VEC and DPM model), excluding the contribution from data loading and preprocessing. Both models were run with a batch size of 1 to emulate the real-world data stream where images are processed as they are received. For comparison, we also show the AIT on a PC for the two models.

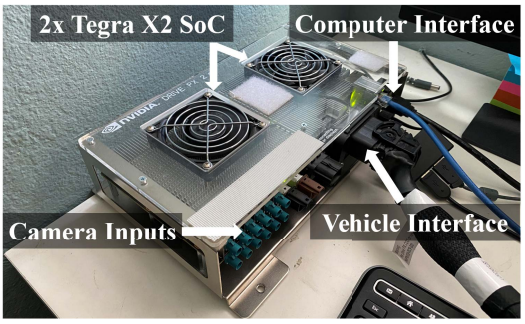

Fig. 6. Our experimental setup for evaluating SG2VEC and DPM on the industry-standard Nvidia DRIVE PX 2 hardware.

TABLE IV
PERFORMANCE EVALUATION OF INFERENCE ON *271-syn* ON THE NVIDIA DRIVE PX 2

Model	PC AIT (ms)	PX2 AIT (ms)	Size (KB)	Power (W)	Energy/frame (mJ)
SG2VEC	**0.2549**	**0.4828**	**331**	**2.99**	**1.44**
DPM	1.393	4.535	2,764	4.42	20.0

Our results are shown in Table IV. SG2VEC performs inference 9.3x faster than the DPM on the PX 2 with an 88.0% smaller model and 32.4% less power, making it undoubtedly more practical for real-world deployment. Our model also uses 92.8% less energy to process each frame, which can be beneficial for electric vehicles with limited battery capacity. With an AIT of 0.4828 ms, SG2VEC can theoretically process up to 2071 frames/second (fps). In contrast, with an AIT of 4.535 ms, the DPM can only process up to 220 fps. In the context of real-world collision prediction, this means that SG2VEC could easily support multiple 60 fps camera inputs from the AV while DPM would struggle to support more than three.

V. CONCLUSION

With our experiments, we demonstrated that our *scene-graph* embedding methodology for collision prediction, SG2VEC, outperforms the state-of-the-art method, DPM, in terms of average MCC (0.5055 versus 0.2096), AIT (0.255 ms versus 1.39 ms), and average time of prediction (39.07% sooner than DPM). Additionally, we demonstrated that SG2VEC could transfer knowledge from synthetic data sets to real-world driving data sets more effectively than the DPM, achieving an average transfer MCC of 0.327 versus 0.060. Finally, we showed that our methodology performs faster inference than the DPM (0.4828 ms versus 4.535 ms) with a smaller model size (331 kB versus 2764 kB) and reduced power consumption (2.99 W versus 4.42 W) on the industry-standard Nvidia DRIVE PX 2 autonomous driving platform. In the context of real-world collision prediction, these results indicate that SG2VEC is a more practical choice for AV safety and could significantly improve consumer trust in AVs. Few works have explored graph-based solutions for other complex AV challenges, such as localization, path planning, and control. These are open research problems that we reserve for future work.

REFERENCES

[1] S. vom Dorff, B. Böddeker, M. Kneissl, and M. Fränzle, "A fail-safe architecture for automated driving," in *Proc. Design Autom. Test Eur. Conf. Exhibit. (DATE)*, 2020, pp. 828–833.

[2] T. Bijlsma *et al.*, "A distributed safety mechanism using middleware and hypervisors for autonomous vehicles," in *Proc. Design Autom. Test Eur. Conf. Exhibit. (DATE)*, 2020, pp. 1175–1180.

[3] "Collision between a sport utility vehicle operating with partial driving automation and a crash attenuator," Nat. Transport. Safety Board, Washington, DC, USA, Rep. NTSB/HAR-20/01, 2020.

[4] "Collision between car operating with partial driving automation and truck-tractor semitrailer," Nat. Transport. Safety Board, Washington, DC, USA, Rep. NTSB/HAB-20/01, 2020.

[5] "Collision between vehicle controlled by developmental automated driving system and pedestrian," Nat. Transport. Safety Board, Washington, DC, USA, Rep. NTSB/HAR-19/03, 2019.

[6] "Global automotive consumer study-North America," Deloitte Develop. LLC, Hermitage, TN, USA, Rep., 2020. [Online]. Available: https://www2.deloitte.com/content/dam/Deloitte/us/Documents/consumer-business/us-global-automotive-consumer-study-2020-north-america.pdf

[7] A. S. Mueller, J. B. Cicchino, and D. S. Zuby, "What humanlike errors do autonomous vehicles need to avoid to maximize safety?" *J. Safety Res.*, vol. 75, pp. 310–318, Dec. 2020.

[8] B. Schoettle and M. Sivak, "A preliminary analysis of real-world crashes involving self-driving vehicles," Univ. Michigan, Transport. Res. Inst., Ann Arbor, MI, USA, Rep. UMTRI-2015–34, 2015.

[9] C. Xu, Z. Ding, C. Wang, and Z. Li, "Statistical analysis of the patterns and characteristics of connected and autonomous vehicle involved crashes," *J. Safety Res.*, vol. 71, pp. 41–47, Dec. 2019.

[10] "Volvo collision avoidance features: Initial results," *Highway Loss Data Inst. Bull.*, vol. 29, no. 5, 2012. [Online]. Available: https://www.iihs.org/media/5d640cd0-fa22-481a-9f6b-4828f15ffa83/-1996076237/HLDI%20Research/Collisions%20avoidance%20features/29.5-Volvo.pdf

[11] J. Dahl, G. R. de Campos, C. Olsson, and J. Fredriksson, "Collision avoidance: A literature review on threat-assessment techniques," *IEEE Trans. Intell. Veh.*, vol. 4, no. 1, pp. 101–113, Mar. 2019.

[12] S. Sontges, M. Koschi, and M. Althoff, "Worst-case analysis of the time-to-react using reachable sets," in *Proc. IEEE Intell. Veh. Symp. (IV)*, 2018, pp. 1891–1897.

[13] J. Nilsson, A. C. E. Ödblom, and J. Fredriksson, "Worst-case analysis of automotive collision avoidance systems," *IEEE Trans. Veh. Technol.*, vol. 65, no. 4, pp. 1899–1911, Apr. 2016.

[14] P. W. Battaglia *et al.*, "Relational inductive biases, deep learning, and graph networks," 2018, *arXiv:1806.01261*.

[15] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," 2017, *arXiv:1711.03938*.

[16] S. Mylavarapu, M. Sandhu, P. Vijayan, K. M. Krishna, B. Ravindran, and A. Namboodiri, "Towards accurate vehicle behaviour classification with multi-relational graph convolutional networks," 2020, *arXiv:2002.00786*.

[17] C. Li, Y. Meng, S. H. Chan, and Y.-T. Chen, "Learning 3D-aware egocentric spatial-temporal interaction via graph convolutional networks," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2020, pp. 8418–8424.

[18] L. Kunze, T. Bruls, T. Suleymanov, and P. Newman, "Reading between the Lanes: Road layout reconstruction from partially segmented scenes," in *Proc. 21st Int. Conf. Intell. Transport. Syst. (ITSC)*, 2018, pp. 401–408.

[19] S. Mylavarapu, M. Sandhu, P. Vijayan, K. M. Krishna, B. Ravindran, and A. Namboodiri, "Understanding dynamic scenes using graph convolution networks," 2020, *arXiv:2005.04437*.

[20] S.-Y. Yu, A. V. Malawade, D. Muthirayan, P. P. Khargonekar, and M. A. Al Faruque, "Scene-graph augmented data-driven risk assessment of autonomous vehicle decisions," *IEEE Trans. Intell. Transp. Syst.*, early access, May 4, 2021, doi: 10.1109/TITS.2021.3074854.

[21] "All New Teslas Are Equipped With NVIDIA's New Drive PX 2 AI Platform For Self-Driving-Electrek." Oct. 2016. [Online]. Available: https://electrek.co/2016/10/21/all-new-teslas-are-equipped-with-nvidias-new-drive-px-2-ai-platform-for-self-driving (accessed 9, Nov. 2020).

[22] S. Shalev-Shwartz, S. Shammah, and A. Shashua, "On a formal model of safe and scalable self-driving cars," 2017, *arXiv:1708.06374*.

[23] D. Nistér, H.-L. Lee, J. Ng, and Y. Wang, "The safety force field," Santa Clara, CA, USA, NVIDIA, Rep., 2019. [Online]. Available: https://www.nvidia.com/content/dam/en-zz/Solutions/self-driving-cars/safety-force-field/the-safety-force-field.pdf

[24] B. Gassmann, F. Pasch, F. Oboril, and K.-U. Scholl, "Integration of formal safety models on system level using the example of responsibility sensitive safety and carla driving simulator," in *Proc. Int. Conf. Comput. Safety Rel. Secur.*, 2020, pp. 358–369.

[25] M. Althoff, O. Stursberg, and M. Buss, "Model-based probabilistic collision detection in autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 2, pp. 299–310, Jun. 2009.

[26] Y. Wang, Z. Liu, Z. Zuo, Z. Li, L. Wang, and X. Luo, "Trajectory planning and safety assessment of autonomous vehicles based on motion prediction and model predictive control," *IEEE Trans. Veh. Technol.*, vol. 68, no. 9, pp. 8546–8556, Sep. 2019.

[27] L. Zhang, W. Xiao, Z. Zhang, and D. Meng, "Surrounding vehicles motion prediction for risk assessment and motion planning of autonomous vehicle in highway scenarios," *IEEE Access*, vol. 8, pp. 209356–209376, 2020.

[28] X. Wang, J. Liu, T. Qiu, C. Mu, C. Chen, and P. Zhou, "A real-time collision prediction mechanism with deep learning for intelligent transportation system," *IEEE Trans. Veh. Technol.*, vol. 69, no. 9, pp. 9497–9508, Sep. 2020.

[29] M. Strickland, G. Fainekos, and H. B. Amor, "Deep predictive models for collision risk assessment in autonomous driving," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2018, pp. 1–8.

[30] B. Asgari, R. Hadidi, N. S. Ghaleshahi, and H. Kim, "PISCES: Power-aware implementation of SLAM by customizing efficient sparse algebra," in *Proc. 57th ACM/IEEE Design Autom. Conf. (DAC)*, 2020, pp. 1–6.

[31] S. Baidya, Y.-J. Ku, H. Zhao, J. Zhao, and S. Dey, "Vehicular and edge computing for emerging connected and autonomous vehicle applications," in *Proc. 57th ACM/IEEE Design Autom. Conf. (DAC)*, 2020, pp. 1–6.

[32] C. Huang, S. Xu, Z. Wang, S. Lan, W. Li, and Q. Zhu, "Opportunistic intermittent control with safety guarantees for autonomous systems," in *Proc. 57th ACM/IEEE Design Autom. Conf. (DAC)*, 2020, pp. 1–6.

[33] J. Yang, J. Lu, S. Lee, D. Batra, and D. Parikh, "Graph R-CNN for scene graph generation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 690–706.

[34] D. Xu, Y. Zhu, C. B. Choy, and L. Fei-Fei, "Scene graph generation by iterative message passing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5410–5419.

[35] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2961–2969.

[36] M. S. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *Proc. Eur. Semantic Web Conf.*, 2018, pp. 593–607.

[37] H. Gao and S. Ji, "Graph U-nets," 2019, *arXiv:1905.05178*.

[38] J. Lee, I. Lee, and J. Kang, "Self-attention graph pooling," 2019, *arXiv:1904.08082*.

[39] V. Ramanishka, Y.-T. Chen, T. Misu, and K. Saenko, "Toward driving scene understanding: A dataset for learning driver behavior and causal reasoning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7699–7707.

[40] Y. Yao, X. Wang, M. Xu, Z. Pu, E. Atkins, and D. Crandall, "When, where, and what? A new dataset for anomaly detection in driving videos," 2020, *arXiv:2004.03044*.

[41] A. P. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," *Pattern Recognit.*, vol. 30, no. 7, pp. 1145–1159, 1997.

[42] D. Chicco and G. Jurman, "The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation," *BMC Genomics*, vol. 21, no. 1, p. 6, 2020.

Arnav Vaibhav Malawade (Graduate Student Member, IEEE) received the B.S. degree in computer science and engineering and the M.S. degree in computer engineering from the University of California at Irvine, Irvine, CA, USA, in 2018 and 2021, respectively, where he is currently pursuing the Ph.D. degree under the supervision of Prof. M. Al Faruque.

His research interests include the design and security of cyber–physical systems in connected/autonomous vehicles, manufacturing, IoT, and healthcare.

Shih-Yuan Yu (Graduate Student Member, IEEE) received the B.S. and M.S. degrees in computer science and information engineering from the National Taiwan University, Taipei, Taiwan, in 2012 and 2014, respectively. He is currently pursuing the Ph.D. degree with the University of California at Irvine, Irvine, CA, USA.

He worked with MediaTek, Hsinchu, Taiwan, for four years. His current research interests are about design automation of embedded systems using data-driven system modeling approaches. It covers incorporating machine learning methods to identify potential security issues in systems.

Brandon Hsu received the B.S. degree in computer engineering from the University of California at Irvine, Irvine, CA, USA, in 2021.

His research interests lie broadly in machine and statistical learning with current focus on perception, representation learning, and autonomous vehicles.

Deepan Muthirayan received the B.Tech./M.Tech. degrees from the Indian Institute of Technology Madras, Chennai, India, in 2010, and the Ph.D. degree from the University of California at Berkeley, Berkeley, CA, USA, in 2016.

He is currently a Postdoctoral Researcher with the Department of Electrical Engineering and Computer Science, University of California at Irvine, Irvine, CA, USA. His doctoral thesis work focused on market mechanisms for integrating demand flexibility in energy systems. Before his term with UC Irvine, he was a Postdoctoral Associate with Cornell University, Ithaca, NY, USA, where his work focused on online scheduling algorithms for managing demand flexibility. His current research interests include control theory, machine learning, topics at the intersection of learning and control, online learning, online algorithms, game theory, and their application to smart systems.

Pramod P. Khargonekar (Life Fellow, IEEE) received the B.Tech. degree in electrical engineering from the Indian Institute of Technology Bombay, Mumbai, India, in 1977, and the M.S. degree in mathematics and the Ph.D. degree in electrical engineering from the University of Florida, Gainesville, FL, USA, in 1980 and 1981, respectively.

He was the Chairman of the Department of Electrical Engineering and Computer Science from 1997 to 2001 and also held the position of a Claude E. Shannon Professor of Engineering Science with The University of Michigan, Ann Arbor, MI, USA. From 2001 to 2009, he was the Dean of the College of Engineering and a Eckis Professor of Electrical and Computer Engineering with the University of Florida till 2016. After serving briefly as the Deputy Director of Technology with ARPA-E, Washington, DC, USA, from 2012 to 2013, he was appointed by the National Science Foundation (NSF) to serve as an Assistant Director for the Directorate of Engineering (ENG) in March 2013, a position he held till June 2016. He is currently the Vice Chancellor for Research and a Distinguished Professor of Electrical Engineering and Computer Science with the University of California at Irvine, Irvine, CA, USA. His research and teaching interests are centered on theory and applications of systems and control.

Dr. Khargonekar has received numerous honors and awards, including the IEEE Control Systems Award, the IEEE Baker Prize, the IEEE CSS Axelby Award, the NSF Presidential Young Investigator Award, and the AACC Eckman Award. He is a Fellow of IFAC and AAAS.

Mohammad Abdullah Al Faruque (Senior Member, IEEE) received the B.Sc. degree in computer science and engineering from Bangladesh University of Engineering and Technology, Dhaka, Bangladesh, in 2002, the M.Sc. degree in computer science from Aachen Technical University, Aachen, Germany, in 2004, and the Ph.D. degree in computer science from Karlsruhe Institute of Technology, Karlsruhe, Germany, in 2009.

He is currently with the University of California at Irvine, Irvine, CA, USA, as an Associate Professor and Directing the Embedded and Cyber–Physical Systems Lab. He served as an Emulex Career Development Chair from October 2012 to July 2015. Before, he was with Siemens Corporate Research and Technology, Princeton, NJ, USA, as a Research Scientist. He has authored two published books. Besides 120+ IEEE/ACM publications in the premier journals and conferences, he holds nine U.S. patents. His current research is focused on the system-level design of embedded and cyber–physical-systems (CPS) with special interest in low-power design, CPS security, and data-driven CPS design.

Dr. Al Faruque was a recipient of the UCI Academic Senate Distinguished Early-Career Faculty Award for Research 2017 and the School of Engineering Early-Career Faculty Award for Research 2017. Besides many other awards, he was a recipient of the School of Engineering Mid-Career Faculty Award for Research 2019, the IEEE Technical Committee on Cyber–Physical Systems Early-Career Award 2018, and the IEEE CEDA Ernest S. Kuh Early Career Award 2016. He is an ACM senior member.