Spatio-Temporal Scene-Graph Embedding for Autonomous Vehicle Collision Prediction

Arnav V. Malawade*, Shih-Yuan Yu*, Brandon Hsu, Deepan Muthirayan, Pramod P. Khargonekar, Mohammad A. Al Faruque
Department of Electrical Engineering & Computer Science, University of California - Irvine, Irvine, CA 92697, USA {malawada, shihyuay, bdhsu, dmuthira, pramod.khargonekar, alfaruqu}@uci.edu

Abstract—In autonomous vehicles (AV), 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 spatio-temporal scene-graph embedding methodology that uses 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 datasets, and 29.47% more accurately on a challenging real-world collision dataset. We also show that SG2VEC is better than the state-of-the-art at transferring knowledge from synthetic datasets to real-world driving datasets. Finally, we demonstrate that SG2VEC performs inference 9.3x 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—scene graph, collision prediction, autonomous vehicles, graph learning, ADAS

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 paper, 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 inter-object relationships [14]. These models also require large datasets that are often costly or unsafe to generate. Synthetic datasets 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 datasets 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.

In summary, the key research challenges associated with autonomous vehicle collision prediction are:

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.
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).

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 inter-object 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 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 inter-object 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 datasets. This paper 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 Figure 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 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 paper 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 datasets and a very challenging real-world collision dataset 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.

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 key to the safety of AVs, a wide range of solutions have 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 like 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 non-trivial and can require vehicle-specific calibration [24].

Model-based probabilistic and deep learning 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. [28] proposes a deep learning collision prediction approach. Still, due to its use of pre-processed 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 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 spatio-temporal 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 dataset, 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 datasets, 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 a multi-relational graph convolutional network (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 Watts 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.

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 spatio-temporal function $f$, where

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

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

SG2VEC consists of two parts (Figure 3): (i) the scene-graph extraction, and (ii) collision prediction through spatio-temporal embedding, described in Section III-A and Section 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, 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. Figure 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 multi-graph, 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 pair-wise 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 inverse perspective mapping (also known as a “birds-eye view” projection) of the image to estimate the locations of objects relative to the ego car, which are used to...
construct the pair-wise relations between objects in $A_n$. For each camera angle, we calibrate the birds-eye view 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 datasets captured by a single vehicle, this step only needs to be performed once. However, for datasets with a wide range of camera angles such as the 620-dash dataset introduced later in the paper, this process needs to be performed once per vehicle. With a human operator, we found that this calibration step takes approximately 1 minute per camera angle on average.

The extraction pipeline identifies three kinds of pair-wise relations: proximity relations (e.g. visible, near, very near, etc.), directional (e.g. Front_Left, Rear_Right, etc.) relations, and belonging (e.g. car_1 is left lane) relations. Two objects are assigned the proximity relation, $r \in \{\text{Near_Collision} (4 \text{ ft.}), \text{Super_Near} (7 \text{ ft.}), \text{Very_Near} (10 \text{ ft.}), \text{Near} (16 \text{ ft.}), \text{Visible} (25 \text{ ft.})\}$ provides the objects are physically separated by a distance that is within that relation’s threshold. The directional relation, $r \in \{\text{Front_Left}, \text{Left_Front}, \text{Left_Rear}, \text{Rear_Left}, \text{Rear_Right}, \text{Right_Front}, \text{Front_Right}\}$, 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_Collision distance from one another. Additionally, the isLeft 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 all vehicles in 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 Figure [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 II-A. Each node $v \in O_n$ is initialized by a one-hot vector (embedding), denoted by $h_v^{(0)}$. Then, the MR-GCN [36] layers are used to update these embeddings via the edges in $A_n$. Specifically, the $l$-th MR-GCN layer computes the node embedding for each node $v$, denoted as $h_v^{(l)}$, as follows:

$$h_v^{(l)} = \Phi_0 \cdot h_v^{(l-1)} + \sum_{r \in A_n} \sum_{u \in N_r(v)} \frac{1}{|N_r(v)|} \Phi_r \cdot h_u^{(l-1)},$$

where $N_v(v)$ denotes the set of neighbors of node $v$ with respect to the relation $r \in A_n$, $\Phi_r$ is a trainable relation-specific transformation for relation $r$, and $\Phi_0$ is the self-connection for each node $v$ that accounts for the influence of $h_u^{(l-1)}$ on $h_v^{(l)}$ [36]. After multiple MR-GCN layers, the output of each layer is concatenated to produce the final embedding for each node, denoted by $H_v^n = \text{CONCAT}(\{h_v^{(l)}\}_{l=0,1,...,L})$, where $L$ is the index of the last layer.

The final embeddings for scene-graph $G_n$, denoted by $X_v^{prop}$, are then passed through a self-attention graph pooling (SAGPooling) layer that filters out irrelevant nodes from the graph, creating the pooled set of node embeddings $X_v^{pool}$ and their edges $A_v^{pool}$. In this layer, we use a graph convolution layer to predict the score $\alpha = \text{SCORE}(X_v^{prop}, A_v^{prop})$ and then use $\alpha$ to perform top-$k$ filtering to filter out the irrelevant nodes in the scene-graph [57].

Then, for each scene-graph $G_n$, the corresponding $X_v^{pool}$ is passed through the graph readout layer that condenses the node embeddings (using operations such as sum, mean, max, etc.) to a single graph embedding $h_{G_n}$. Then, this spatial embedding $h_{G_n}$ is passed to the temporal model (LSTM) to generate a spatio-temporal embedding $z_n$ as follows:

$$z_n, s_n = \text{LSTM}(h_{G_n}, s_{n-1}) \quad (3)$$

The hidden state $s_n$ of the LSTM is updated for each times-tamp $n$. Lastly, each spatio-temporal embedding $z_n$ is then passed through a Multi-Layer Perceptron (MLP) that outputs each class’s confidence value. The two outputs of the MLP are compared, and $Y_n$ 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 non-binarized outputs $Y_n$ 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 2 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 64x64x5 Convolutional LSTM (ConvLSTM) layer, one
32x32x5 ConvLSTM layer, one 16x16x5 ConvLSTM layer, one MLP layer with output size 64, one MLP layer with output size 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. Dataset Preparation

We prepared three types of datasets for our experiments: (i) synthesized datasets, (ii) a typical real-world driving dataset, and (iii) a complex real-world driving dataset. Examples from each dataset are shown in Figure 2. Our synthetic datasets focus on the highway lane change scenario as it is a common AV task. To evaluate the transferability of each model from synthetic datasets to real-world driving, we prepared a typical real-world dataset containing lane-change driving clips. Finally, we prepared the complex real-world driving dataset to evaluate each model’s performance on a challenging dataset containing a broad spectrum of collision types, road conditions, and vehicle maneuvers. All datasets were collected at a 1280x720 resolution, and each clip spans 1-5 seconds.

1) Synthetic Datasets: To synthesize the datasets, we developed a tool using CARLA [15], an open-source driving simulator, and CARLA Scenario Runner [16] 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 datasets: a 271-syn dataset and a 1043-syn dataset, containing 271 and 1,043 video clips, respectively. These datasets have no-collision:collision label distributions of 6.12:1 and 7.91:1, respectively. In addition, we sub-sampled the 1043-syn dataset to create 306-syn: a balanced dataset that has a 1:1 distribution. Our synthetic scene-graph dataset [17] and our source code [18] are open-source and available online.

2) Typical Real-World Driving Dataset: This dataset, denoted as 571-honda, is a subset of the Honda Driving Dataset (HDD) [38] 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 Dataset: Our complex real-world driving dataset, denoted as 620-dash, contains very challenging real-world collision scenarios drawn from the Detection of Traffic Anomaly dataset [39]. This dataset contains a wide range of drivers, car models, driving maneuvers, weather/road conditions, and collision types, as recorded by on-board dashboard cameras. Since the original dataset contains only collision clips, we prepared 620-dash by splitting each clip in the original dataset into two parts: (i) the beginning of the clip until 1 second before the collision, and (ii) from 1 second before the collision until the end of the collision. We then labeled part (i) as ‘no-collision’ and part (ii) as ‘collision.’ The 620-dash dataset contains 315 collision video clips and 342 non-collision driving clips.

4) Labeling and Pre-Processing: We labeled the synthetic datasets and the 571-honda dataset 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 datasets were pre-processed 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 datasets were pre-processed to match the input format used in its original implementation [29]. Thus, the DPM uses 64x64 grayscale versions of the clips in the datasets turned into sets of subsequences Jn for a clip of length l defined as follows.

\[ J_n = \{ I_n, I_{n+1}, I_{n+2}, I_{n+3}, I_{n+4} \}, \text{for } n \in [1, l - 4] \]  

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) [40], and Matthews Correlation Coefficient (MCC) [41]. MCC is considered a balanced measure of performance for binary classification even on datasets 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 re-weighting helps compensate for the dataset imbalance during training, classification accuracy is typically less reliable for imbalanced datasets, so the primary metric we use to compare the models is MCC. We used stratified 5-fold cross-validation to produce the final results shown in Table I and Figure 5.

![https://github.com/carla-simulator/scenario_runner](https://github.com/carla-simulator/scenario_runner)
![https://dx.doi.org/10.21227/c0z9-1p30](https://dx.doi.org/10.21227/c0z9-1p30)
![https://github.com/AICPS/sg-collision-prediction](https://github.com/AICPS/sg-collision-prediction)

| Dataset       | Model     | Accuracy | AUC    | MCC     |
|---------------|-----------|----------|--------|---------|
| 271-syn       | SG2VEC (5-frames) | 0.8795   | 0.9541 | 0.5362  |
| 271-syn       | SG2VEC    | 0.8512   | 0.8439 | 0.2160  |
| 271-syn       | DPM       | 0.7373   | 0.6853 | 0.5790  |
| 1043-syn      | SG2VEC (5-frames) | 0.9142   | 0.9623 | 0.5223  |
| 1043-syn      | SG2VEC    | 0.9095   | 0.9477 | 0.5385  |
| 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 |

TABLE I: Classification accuracy, AUC, and MCC for SG2VEC (Ours) and DPM.
1) Synthetic Datasets: The performance of SG2VEC and the DPM on our synthetic datasets is shown in Table I. We find that our SG2VEC achieves higher accuracy, AUC, and MCC on every dataset, 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 datasets and slightly lower overall performance on the balanced datasets. This is likely because the large number of safe lane changes in the imbalanced datasets 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 performs better on the balanced datasets.

The DPM achieves relatively high accuracy and AUC on the imbalanced 271-syn and 1043-syn datasets, but suffers significantly on the balanced 306-syn dataset. This drop indicates that the DPM could not identify the minority class (collision) well and tended to over-predict the majority class (no-collision). In terms of MCC, the DPM scores higher on the 306-syn dataset than what it scores on the other datasets. This result is because the 306-syn dataset has a balanced class distribution compared to the other datasets, 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 datasets 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.

2) Complex Real-World Dataset: The performance of both the models significantly drops on the highly complex real-world 620-dash dataset due to the variations in the driving scenes and collision scenarios. This drop is to be expected as this dataset 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 birds-eye view (BEV) calibration on this dataset in comparison to the other datasets. Since the varying camera angles and road conditions in this dataset impact our ability to properly calibrate SG2VEC’s BEV projection in a single step, we created custom BEV calibrations for each clip in the dataset, which improved performance somewhat. However, as shown in Figure 4c, a significant part of the dataset 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.

The results from our SG2VEC ablation study are shown in Table III and support our hypothesis that both spatial modeling with MRGCN and temporal modeling with LSTM are core to 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.
TABLE II: Average time of prediction (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 |

TABLE III: sg2VEC ablation study on the 1043-syn dataset.

![Fig. 5: Performance after transferring the models trained on synthetic 271-syn and 1043-syn datasets to the real-world 571-honda dataset.](image)

C. Transferability From Synthetic to Real-World Datasets

The collision prediction models trained on simulated datasets 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 dataset before testing it on the 571-honda dataset. No additional domain adaptation was performed. We did not evaluate transferability to the 620-dash dataset because it contains a wide range of highly dynamic driving maneuvers that were not present in our synthesized datasets. As such, evaluating transferability between our synthesized datasets and the 620-dash dataset would yield poor performance and would not provide insight. Figure 5 compares the accuracy and MCC for both the models on each training dataset and the 571-honda dataset 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 dataset 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 dataset can be attributed to the characteristic differences between the simulated and real-world datasets; the 571-honda dataset 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 dataset drops more than the model trained on the 1043-syn dataset after the transfer, likely due to the smaller training dataset size. Regarding accuracy, the sg2VEC model trained on 1043-syn achieves 4.37% higher accuracy and the model trained 271-syn dataset achieves 1.47% lower accuracy than the DPM model trained on the same datasets. The DPM’s similar accuracy after

---

high learning rate and batch size, we achieved significantly better performance than the model architecture discussed in the beginning of Section IV (2 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 dataset 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 (1 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. Model performance on this dataset and similar datasets 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 datasets 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 dataset is worse than that of a random classifier (MCC of 0.0). Consistent with the synthetic dataset results, sg2VEC using all frames performs better on the balanced 620-dash dataset 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.

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 II. On the 1043-syn dataset, 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 dataset represents 1.867 seconds of data. Thus, our methodology predicted collisions **122.7** milliseconds earlier than DPM on average. This extra time can be critical for ensuring that the AV avoids an impending collision.
transfer likely results from the class imbalance in the 571-honda dataset. 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 deep learning 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 Figure 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 SG

\[
\text{VECTOR}'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 deep learning 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 Figure 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 pre-processing. 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.

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 2,071 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.

### 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      | 0.8028       | 331       | 2.99      | 1.44               |
| DPM     | 4.093       | 4.535        | 2,764     | 4.42      | 20.0               |

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 vs. 0.2096), average inference time (0.255 ms vs. 1.39 ms), and average time of prediction (39.07% sooner than DPM). Additionally, we demonstrated that SG2VEC could transfer knowledge from synthetic datasets to real-world driving datasets more effectively than the DPM, achieving an average transfer MCC of 0.327 vs. 0.060. Finally, we showed that our methodology performs faster inference than the DPM (0.4828 ms vs. 4.535 ms) with a smaller model size (331 KB vs. 2.764 KB) and reduced power consumption (2.99 W vs. 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. Büddeke, et al., “A fail-safe architecture for automated driving,” 2020 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 828–833, 2020.

[2] T. Bijlsma, A. Buriachevskyi, et al., “A distributed safety mechanism using middleware and hypervisors for autonomous vehicles,” 2020 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 1175–1180, 2020.

[3] National Transportation Safety Board, “Collision Between a Sport Utility Vehicle Operating With Partial Driving Automation and a Crash Attenuator,” National Transportation Safety Board, Tech. Rep. NTSB/HAR-20/01, 2020.

[4] ———, “Collision Between Car Operating with Partial Driving Automation and Truck-Tractor Semitrailer,” National Transportation Safety Board, Tech. Rep. NTSB/HAB-2001, 2020.

[5] ———, “Collision between vehicle controlled by developmental automated driving system and pedestrian,” National Transportation Safety Board, Tech. Rep. NTSB/HAR-19/03, 2019.

[6] Deloitte Development LLC, “Global Automotive Consumer Survey - North America,” Deloitte Development LLC, Tech. Rep., 2020.

[7] A. S. Mueller, J. B. Cicchino, and D. S. Zuby, “What humanlike errors do autonomous vehicles need to avoid to maximize safety?”, Journal of Safety Research, 2020.

[8] B. Schoettle and M. Sivak, “A preliminary analysis of real-world crashes involving self-driving vehicles,” University of Michigan Transportation Research Institute, Tech. Rep. UMTRI-2015-34, 2015.

[9] C. Xu, Z. Ding et al., “A fail-safe architecture for automated driving,” in 2020 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 828–833, 2020.

[10] ———, “Collision Between Vehicle Controlled by Developmental Automation and Truck-Tractor Semitrailer,” National Transportation Safety Board, Tech. Rep. NTSB/HAB-2001, 2020.

[11] ———, “A distributed safety mechanism using middleware and hypervisors for autonomous vehicles,” Journal of safety research, vol. 71, pp. 41–47, 2019.

[12] H. L. D. Institute, “Volvo collision avoidance features: initial results,” Highway Loss Data Institute Bulletin, vol. 29, no. 5, 2012.

[13] J. Dahl, G. R. de Campos et al., “Collision avoidance: A literature review on threat-assessment techniques,” IEEE Transactions on Intelligent Vehicles, vol. 4, no. 1, pp. 101–113, 2018.

[14] S. Sontges, M. Koschel, and M. Althoff, “Worst-case analysis of the time-to-react using reachable sets,” in 2018 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2018, pp. 1891–1897.
Pramod Khargonekar received B. Tech. Degree in electrical engineering in 1977 from the Indian Institute of Technology, Bombay, India, and M.S. degree in mathematics in 1980 and Ph.D. degree in electrical engineering in 1981 from the University of Florida, respectively. He was Chairman of the Department of Electrical Engineering and Computer Science from 1997 to 2001 and also held the position of Claude E. Shannon Professor of Engineering Science at The University of Michigan. From 2001 to 2009, he was Dean of the College of Engineering and Eckis Professor of Electrical and Computer Engineering at the University of Florida till 2016. After serving briefly as Deputy Director of Technology at ARPA-E in 2012-13, he was appointed by the National Science Foundation (NSF) to serve as Assistant Director for the Directorate of Engineering (ENG) in March 2013, a position he held till June 2016. Currently, he is Vice Chancellor for Research and Distinguished Professor of Electrical Engineering and Computer Science at the University of California, Irvine. His research and teaching interests are centered on theory and applications of systems and control. He has received numerous honors and awards including IEEE Control Systems Award, IEEE Baker Prize, IEEE CSS Axelby Award, NSF Presidential Young Investigator Award, AACC Eckman Award, and is a Fellow of IEEE, IFAC, and AAAS.

Mohammad Abdullah Al Faruque (M’06, SM’15) received his B.Sc. degree in Computer Science and Engineering (CSE) from Bangladesh University of Engineering and Technology (BUET) in 2002, and M.Sc. and Ph.D. degrees in Computer Science from Aachen Technical University and Karlsruhe Institute of Technology, Germany in 2004 and 2009, respectively. He is currently with the University of California Irvine (UCI) as an Associate Professor and Directing the Embedded and Cyber-Physical Systems Lab. He served as an Emulex Career Development Chair from October 2012 till July 2015. Before, he was with Siemens Corporate Research and Technology in Princeton, NJ as a Research Scientist. 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, data-driven CPS design, etc. He is an ACM senior member. He is the author of 2 published books. Besides many other awards, he is the 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 also the 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 120+ IEEE/ACM publications in the premier journals and conferences, he holds 9 US patents.