RS2G: Data-Driven Scene-Graph Extraction and Embedding for Robust Autonomous Perception and Scenario Understanding

Junyao Wang*, Arnav Vaibhav Malawade*, Junhong Zhou,
Shih-Yuan Yu, Mohammad Abdullah Al Faruque
University of California, Irvine
Irvine, United States, 92697
{junyaow4, malawada, junhonz2, shihyuay, alfaruqu}@uci.edu

Abstract

Effectively capturing intricate interactions among road users plays a critical role in achieving safe navigation for autonomous vehicles. While graph learning (GL) has emerged as a promising approach to tackle this challenge, existing GL models rely on predefined domain-specific graph extraction rules and often fail in real-world dynamic scenarios. Additionally, these graph extraction rules severely impede the capability of existing GL methods to generalize knowledge across domains. To address this issue, we propose ***RoadScene2Graph (RS2G)****, an innovative autonomous scenario understanding framework with a novel data-driven graph extraction and modeling approach that dynamically captures the diverse relations among road users. Our evaluations show that on average RS2G outperforms the state-of-the-art (SOTA) rule-based graph extraction method by* 4.47% *and the SOTA deep learning model by* 22.19% *in subjective risk assessment. RS2G also delivers notably better performance in transferring knowledge gained from simulations to unseen real-world scenarios.*

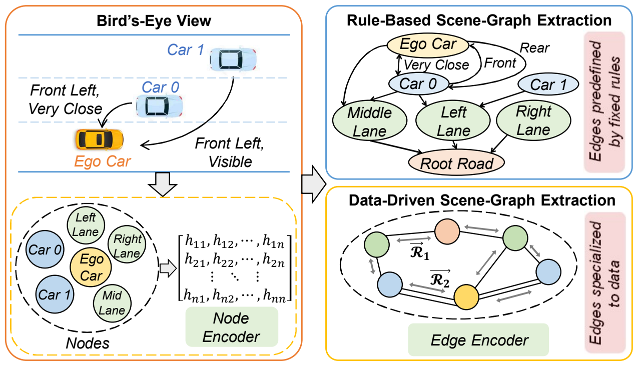

Figure 1. Both rule-based and data-driven scene-graph extraction methods start with transforming objects to nodes with a node encoder, where each node has an attribute vector $\mathbf{h}$. The rule-based scene-graph extraction relies on fixed rules derived from expert knowledge; its encoded edges typically have concrete physical meanings and the graphs are constrained by specific domains. Our data-driven scene-graph extraction represents diverse relations between nodes with vectors, e.g. $\mathcal{R}_1, \mathcal{R}_2$, which better captures latent features and can be more dynamic and domain-adaptive.

1. Introduction

Ensuring road safety to support various driving conditions has emerged as a fundamental research topic in the domain of *autonomous vehicles* (AVs) [34, 44]. As human drivers naturally reason about interactions between road users to effectively navigate their environments [35], a number of innovative works modeling human driving experience to enhance the safety and robustness of AVs have been proposed [8, 9, 15, 25, 45]. However, it remains challenging to understand and model the diverse relations among driving agents to achieve adequate performance for autonomous risk assessment [16, 17]. Additionally, AVs are typically trained and tested with simulations and synthetic data, while real-world scenarios are often much more dynamic and unpredictable. This necessitates a learning approach that (i) adaptively captures and models interactions between road users and environments, and (ii) effectively transfers knowledge from training domains to real-world settings.

Many existing works leverage deep learning (DL) approaches, e.g., convolution neural networks (CNNs), to model human driving capabilities [8, 9, 11]. However, these methods often fail to account for information of high-level semantic scenes, thus performing less well in more complex or novel scenarios. Specifically, they rarely consider interactions between driving agents and environmental factors, e.g., effects of traffic signals on human behaviors [2, 5, 11, 40]. Additionally, although existing DL-based data-driven approaches have demonstrated enhanced capabilities to generalize models across domains, i.e., improving the *robustness* of models, these approaches often involve extraordinarily sophisticated models and massive labeled datasets [10, 42]. Thus, these approaches can be extremely expensive and hence impractical as AVs are real-time systems with limited computational resources and onboard energy storage [7, 24]. Besides, existing datasets are typically biased toward everyday driving situations but not diverse corner cases, often resulting in inadequate model performance in scenarios involving higher risks [9, 16, 46].

* Equal contribution.

In contrast to CNN-based models that directly extract visual features, graph learning (GL) has emerged as a promising approach to explicitly capturing high-level interactions between visual features [4, 23, 32]. Prior works have shown that graph representations of driving scenarios extracted based on domain knowledge, referred to as rule-based *scene-graphs*, enables effective modeling of diverse relations among road users and can potentially enhance autonomous scenario understanding [19, 29, 36]. As demonstrated in Figure 1, nodes of a rule-based scene-graph represent objects in a scene, e.g., lanes, vehicles, and traffic lights, while edges represent the types of relations, e.g., near and front left. Additionally, it has been observed that sence-graphs can considerably improve data efficiency and transfer learning at AV safety-related tasks, e.g., collision prediction [26, 40]. Unfortunately, existing scene-graph extraction relies on predefined domain-specific rules, e.g., rule-based distance relations and road topology [19, 29, 36, 40], resulting in rigid graphical structures. These graph extraction rules often fail to provide graph representations that are sufficiently expressive to achieve adequate performance. The effectiveness of these rules also varies widely across domains, severely limiting their capability to generalize to real-world scenarios absent from the training data.

To address this issue, we propose ***RoadScene2Graph (RS2G)***, an innovative autonomous scenario understanding framework with a novel data-driven graph extraction and modeling approach that *dynamically* captures the diverse relations among road users. As demonstrated in Figure 1, in contrast to rule-based graph extraction methods, RS2G represents graph edges with vectors that reflect the probability distribution of different object relations and capture the relations between nodes in a more granular manner. The resulting graph representation hence becomes more expressive and *domain-adaptive*. Specifically, we construct relations between nodes leveraging the Transformer as the edge encoder, since it has been proven that the attention mechanism of the Transformer is particularly powerful at capturing dependencies within inputs [37]. We also utilize the variational autoencoder (VAE) to further improve the expressiveness of our graph representations and provide enhanced autonomous scenario understanding. The main contributions of our paper are listed as follows:

- We introduce RS2G, an innovative autonomous driving risk assessment framework with a novel data-driven scene-graph extraction and modeling method. RS2G dynamically learns node embeddings and extracts the optimal graph representation of a road scene.
- To the best of our knowledge, RS2G is the first graph extraction method leveraging the powerful attention mechanism of the Transformer to capture relations and dependencies among road users. Our graph extraction technique significantly enhances model performance for both subjective risk assessment and transfer learning from simulations to real-world scenarios.
- We conduct detailed ablation studies regarding the benefits of each component of RS2G and further demonstrate the advantage of our data-driven graph extraction method in real-world autonomous scenario understanding.

2. Related Works

2.1. Interaction Modeling for Autonomous Driving

Several recent works have demonstrated that explicitly modeling interactions between agents in dynamic environments can improve autonomous systems' capability to understand and reason about their environment [21, 46]. Multiple innovative learning frameworks using domain knowledge to extract graph representations of driving scenarios, i.e., *scene-graphs*, for AVs have also been proposed. In particular, [36] proposes a rule-based graph extraction method encoding relationships between road users, and shows that this graph representation enables an effective autoencoder for inferring relationships between road users in unseen scenarios. [19] combines rule-based scene-graph extraction and MR-GCN to provide explainable predictions of future driver actions, and [40] shows how a rule-based scene-graph improves risk assessment performance over CNN-based methods. Besides, [29] uses a rule-based graph extracted from multi-modal sensor data to perform accurate driver action prediction. Unfortunately, these methods rely on domain knowledge and rule-based graph extraction techniques, restricting them to their specialized tasks and data domains. In other words, different tasks require defining new rules, and each aforementioned work has a different set of rules. In contrast, our data-driven graph extraction approach eliminates such overhead, as we learn the graph extraction rules directly from the data, enabling high performance across tasks and data domains.

2.2. Transfer Learning for Autonomous Driving

Generalizing a trained model to unseen real-world scenarios without substantial performance degradation remains a critical challenge in autonomous driving. The term *Sim2Real* describes the capability of a robotic system to transfer knowledge gained from simulation environments to real-world applications [14]. Existing approaches addressing the transfer learning challenge can be mainly categorized as *inductive* transfer learning and *transductive* transfer learning [30]. Inductive transfer learning involves learning a general set of rules from the source domain, e.g., training a supervised learning model, and applying them to the test domain. In contrast, transductive transfer learning utilizes some knowledge of the test domain to adapt a model trained in the source domain. Our work focuses on the inductive case as it better aligns with typical autonomous system applications, i.e., training machine learning (ML) models with processed or simulated data and testing them in diverse real-world settings. Several prior studies have leveraged inductive transfer learning to enhance model generalization capabilities. [43] transfers a CNN-based motion prediction model trained on pedestrian/vehicle trajectories to cyclist trajectories, and demonstrates that transferring knowledge from pedestrian motion prediction improves the performance of the cyclist motion prediction. [20] transfers knowledge from semi-supervised models with contrastive learning and teacher-student methods to improve trajectory prediction performance, and [27] evaluates transfer learning from traditional camera models to event camera models for steering angle prediction. [1] transfers spatial-temporal features and uses salient data augmentation for better Sim2Real transfer performance in steering angle prediction and collision detection. More recently, [40] demonstrates that graph-based scene modeling improves Sim2Real transfer performance compared to CNN-based methods, while its domain-specific graph extraction rules can limit the model adaptivity across domains.

3. Methodology

3.1. Problem Formulation

The problem of subjective risk assessment starts with a sequence of sensor data pre-processed by an object detection model and converted to a set of scene-graphs. These scene-graphs are then transformed into spatial-temporal embeddings for the subjective risk assessment. The overall system can be modeled as a binary classification task. Specifically, given that the input $\mathcal{I}$ is a sequence of sensor data, e.g., camera images, of length $\mathcal{T}$, a model Φ is constructed to generate the output $\mathcal{Y}$, i.e.,

$$\mathcal{Y} = \Phi(\mathcal{I}); \quad \mathcal{I} = \{i_1, i_2, ..., i_{\mathcal{T}}\} \tag{1}$$

$$\text{and} \quad \mathcal{Y} = \begin{cases} 0, & \text{if the driving sequence is safe} \\ 1, & \text{if the driving sequence is risky} \end{cases}$$

where $\mathcal{Y}$ denotes the subjective risk indicator of the driving scene and Φ represents the function mapping inputs $\mathcal{I}$ to $\mathcal{Y}$. An ML model is often utilized to approximate the function defined by Φ, where the inference output is denoted as $\hat{\mathcal{Y}}$.

Some ML methods, namely CNNs, directly operate on sensor inputs to produce output classifications $\hat{\mathcal{Y}}$. However, these approaches only model pixel-level features without considering inter-object relations for high-level objectives. On the other hand, CNN-based models typically perform well for low-level tasks, such as object detection, since these tasks are less dependent on inter-object semantic relations. Therefore, for each input data sample i_t in the sequence, i.e., $i_t \in \mathcal{I}$, we first utilize a pre-trained CNN-based object detection model Ω to efficiently extract the set of objects $\mathcal{O}_t$ and their attributes $\mathcal{D}_t$. We then utilize our graph extraction model Ψ (elaborated in section 3.2) to generate a scene-graph $\mathcal{G}_t$, i.e.,

$$\mathcal{O}_t, \mathcal{D}_t = \Omega(i_t) \text{ and } \mathcal{G}_t = \Psi(\mathcal{O}_t, \mathcal{D}_t) \tag{2}$$

We denote a *scene-graph* as $\mathcal{G}_t = (\mathcal{O}_t, \mathcal{A}_t)$ and model it as a directed heterogeneous multi-graph since multiple types of edges can exist between any two nodes. $\mathcal{O}_t$ denotes the set of nodes and represents objects in a scene. The edges of $\mathcal{G}_t$ are represented by the adjacency matrix $\mathcal{A}_t$, and each value in $\mathcal{A}_t$ represents the type of the corresponding edge in $\mathcal{G}_t$. Once all the scene-graphs are extracted for the current scene, we analyze the collection of graphs $\mathcal{G}$ with our spatial-temporal graph embedding model Φ (elaborated in section 3.3) to classify $\mathcal{Y}$ by its risk. Thus, the complete system can be modeled as

$$\hat{\mathcal{Y}} = \Phi(\mathcal{G}) \tag{3}$$

where $\mathcal{G} = \{\Psi(\Omega(i_t)) \; \forall t \in \{1, 2, ..., \mathcal{T}\}\}$.

The architecture of our proposed RS2G is demonstrated in Figure 2. We elaborate on each component of RS2G in the rest of this section.

3.2. Data-Driven Scene-Graph Extraction

Our methodology for extracting a scene-graph $\mathcal{G}_t$ from a set of objects $\mathcal{O}_t$ and their attributes $\mathcal{D}_t$ at a given time t is demonstrated in Algorithm 1. In contrast to the SOTA graph extraction method [40] using fixed rules derived from domain knowledge to construct graph edges, e.g., threshold-based distance relations and compass-based directional relations, we propose an innovative data-driven edge encoder to generate *domain-specialized* edge types. Specifically, our approach starts with a node encoder model $Encode_{node}$ that converts the attributes $\mathbf{d}_j(\mathbf{d}_j \in \mathcal{D}_t)$ of each object $o_j(o_j \in \mathcal{O}_t)$ into a set of encoded node features $\mathbf{h}_j \in \mathcal{H}_t$ by its entity type, e.g., car and lane, and coordinate, i.e. location, where $\mathcal{H}_t$ denotes the collection of encoded node features of all the objects at time t. We then concatenate feature vectors of each pair of nodes and send the resulting vector to an edge encoder, denoted as $Encode_{edge}$ in Algorithm 1, to infer if there is an edge of a given relation type $r \in \mathcal{R}$, given their features $\mathbf{h}_j$ and $\mathbf{h}_k$. Here $\mathcal{R}$ denotes the set of all the possible relations between nodes, and (j,k) represents all the possible combinations of two nodes in $\mathcal{H}$, i.e., $\{(j,k) | \forall j,k \in \mathcal{H}, j \neq k\}$. Each relation type has a different set of learnable weights, and our edge encoder is responsible for computing these weights to find different rules for constructing each relation type. Intuitively, with a strong ability to capture complicated non-linear mappings and hierarchical feature interactions, a multi-layer perceptron (MLP) can effectively discern relations between nodes and thereby serve as a proper edge encoder.

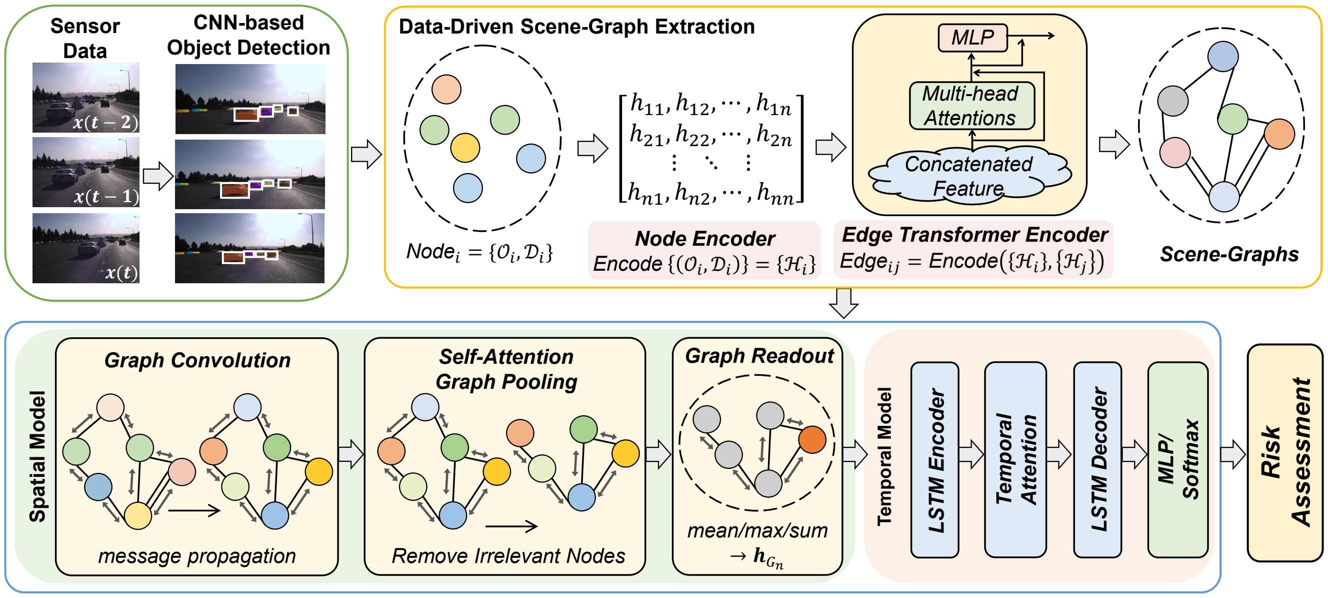

Figure 2. The Architecture of Our Proposed RS2G. Given an input data sample, RS2G starts with a set of objects and their attributes extracted by a pre-trained CNN-based model. We then utilize our data-driven scene-graph extraction to generate a set of scene-graphs of the current scene and analyze it with our spatial-temporal embedding model. Finally, we utilize a multi-layer perceptron (MLP) to classify the risk of the driving scenario as risky or non-risky.

Algorithm 1: Data-Driven Scene-Graph Extraction

```
1  Input: Objects $\mathcal{O}_t$ and their attributes $\mathcal{D}_t$ at time $t$.
2  Output: Scene-graph $\mathcal{G}_t$ at time $t$.
3  def $\Psi(\mathcal{O}_t, \mathcal{D}_t)$:
4      $\mathcal{H}_t \leftarrow \emptyset, \mathcal{A}_t \leftarrow \mathbf{0}_{n\times n}$          ▷ initialize outputs
5      for $o_j, \mathbf{d}_j \in \mathcal{O}_t, \mathcal{D}_t$ do
6          $\mathbf{h}_j \leftarrow Encode_{node}(o_j, \mathbf{d}_j)$          ▷ node encoding
7          $\mathcal{H}_t.append(\mathbf{h}_j)$
8      $\mathcal{C} \leftarrow \mathcal{H}_t \times \mathcal{H}_t$          ▷ get all pair of nodes
9      for relation $r \in \mathcal{R}$ do
10         for edge $(\mathbf{h}_j, \mathbf{h}_k) \in \mathcal{C}$ do
11             $(\mathcal{A}_t)_{r,j,k} \leftarrow MLP(Encode_{edge}(r, \mathbf{h}_j, \mathbf{h}_k))$
12     $\mathcal{G}_t \leftarrow \{\mathcal{H}_t, \mathcal{A}_t\}$
13     return $\mathcal{G}_t$
```

3.2.1 Edge Encoder Inspired by the Transformer

To further enhance RS2G's capability to model complex relations among road users, we propose an innovative edge encoder based on the Transformer to process the concatenated features. Transformer is an attention-based model that performs particularly well in capturing dependencies among input vectors, and naturally fits this edge prediction task that aims to model relationships between nodes. Specifically, we first calculate the transformed information $\mathcal{Q}$ (query), $\mathcal{K}$ (key), and $\mathcal{V}$ (value) representations from node features as $\mathbf{h}_t\mathcal{W}_i^{\mathcal{Q}}, \mathbf{h}_t\mathcal{W}_i^{\mathcal{K}}, \mathbf{h}_t\mathcal{W}_i^{\mathcal{V}}$, where $\mathbf{h}_t$ represents the feature vector of a node at time t and $\mathcal{W}_i^{\mathcal{Q}}$, $\mathcal{W}_i^{\mathcal{K}}$, $\mathcal{W}_i^{\mathcal{V}}$ are trainable parameters for $\mathcal{Q}, \mathcal{K}, \mathcal{V}$, respectively. We then utilize an attention mechanism to enable each node to understand its context and implicitly establish relationships with other nodes in the scene. We obtain the relation representation τ by:

$$\tau = \mathcal{Q}\mathcal{K}^T\mathcal{V} \tag{4}$$

Finally, we apply a multi-layer perceptron (MLP) to transform τ from dimension $\mathbf{h}_j$ to the total number of potential relations $|\mathcal{R}|$. The output of our data-driven scene-graph extraction is an $n \times n \times |\mathcal{R}|$ adjacency matrix $\mathcal{A}_t$, where n denotes the total number of nodes and $|\mathcal{R}|$ denote the number of relation types. Along with the node features $\mathcal{H}_t$, this adjacency matrix $\mathcal{A}_t$ forms the backbone of the scene-graph $\mathcal{G}_t$, offering a more dynamic representation of scenes.

3.2.2 Scene-Graph Generalization

In the context of autonomous driving, we aim to enable our model to comprehensively capture all the essential relations among road users while avoiding excessively including minor details; otherwise, it can cause overfitting and compromise the generalization capability of our model. Considering the powerful nature of the Transformer model in capturing intricate dependencies and relations among inputs, instead of directly using the adjacency matrix $\mathcal{A}_t$ as graph representations, we introduce the variational autoencoder (VAE) for regularization to enhance our model's generalization capabilities across various driving conditions. Specifically, we reparameterize each relation vector representing edge types in $\mathcal{A}_t$ as

$$z = \mu + \log(\sigma^2) \times \epsilon \tag{5}$$

where z represents a new relation vector transformed from regularizing each relation vector in $\mathcal{A}_t$; μ and σ^2 indicate the mean and variance of each relation vector in $\mathcal{A}_t$, respectively, and ϵ represents random samples from a standard normal distribution, i.e., $\epsilon \sim \mathcal{N}(0, 1)$. This serves as a "bottleneck" that enforces the model to capture the most salient features of the adjacency matrix. Specifically, this reparameterization technique enables a back-propagation by incorporating more randomness into the model, hence ensuring a smooth gradient landscape for optimization. Moreover, we involve the Kullback-Leibler (KL) divergence as a loss term to train our model. It not only ensures a more meaningful structure of the latent space that provides a more generalized and effective relation representation, but also enables the model to generate more expressive adjacency matrices to enhance graph representations of road scenes.

3.3. Spatial-Temporal Graph Embedding Model

As demonstrated in Figure 2, our spatial-temporal model consists of three major components, a spatial model, a temporal model, and a risk inference component. The spatial model outputs the embeddings $\mathbf{h}_{\mathcal{G}_t}$ for each scene-graph $\mathcal{G}_t$, and the temporal model processes all input scene-graph embeddings, i.e., $\{\mathbf{h}_{\mathcal{G}_1}, \mathbf{h}_{\mathcal{G}_2}, \ldots, \mathbf{h}_{\mathcal{G}_\mathcal{T}}\}$, and produces the spatial-temporal embedding $\mathcal{Z}$. Then the risk inference components output each driving clip's risk assessment $\hat{\mathcal{Y}}$.

3.3.1 Spatial Graph Modeling

We utilize a multi-relational graph convolutional network (MR-GCN) [28] to compute the embeddings and capture multiple types of relations on each scene-graph. Specifically, in the *message propagation* phase, each MR-GCN layer performs spatial graph convolutions [18] on each graph $\mathcal{G}_t = \{\mathcal{H}_t, \mathcal{A}_t\}$ for all $t \in \{1, 2, \ldots, \mathcal{T}\}$ across a set of relation types $\mathcal{R}$, where $\mathcal{T}$ denote the length of the data sample. For each node $v \in \mathcal{G}_t$, the l-th MR-GCN layer updates the node embedding, denoted as $\mathbf{h}_v^{(l)}$, as

$$\mathbf{h}_v^{(l)} = \mathbf{\Phi}_0 \cdot \mathbf{h}_v^{(l-1)} + \sum_{r \in \mathcal{R}} \sum_{u \in \mathbf{N}_r(v)} \frac{1}{|\mathbf{N}_r(v)|} \mathbf{\Phi}_r \cdot \mathbf{h}_u^{(l-1)}, \tag{6}$$

where $\mathbf{N}_r(v)$ denotes the set of neighbor indices to v with relation r in the adjacency matrix $\mathcal{A}_t$, $\mathbf{\Phi}_r$ represents the set of trainable weights for relation r in MR-GCN layer l. Since the $(l-1)$-th layer can directly influence the node representations in the l-th layer, MR-GCN applies another trainable transformation Φ_0 to account for the self-connection of each node using a special relation [33]. We initialize each node embedding $\mathbf{h}_v^{(0)}, \forall v \in \mathcal{O}_t$ by directly converting the node's type information to its corresponding one-hot vector.

Node embeddings typically become more refined and abstract as the number of MR-GCN layers increases, while the output of the features from earlier MR-GCN layers can be better generalized across domains [38]. Therefore, we utilize the node embeddings generated from all the MR-GCN layers. Specifically, we calculate the embedding of node v at the final layer, denoted as $\mathcal{H}_v^{\mathcal{L}}$, by concatenating the features generated from all the MR-GCN layers, i.e.,

$$\mathcal{H}_v^{\mathcal{L}} = \mathbf{CONCAT}(\{\mathbf{h}_v^{(l)}\} | l = 0, 1, ..., \mathcal{L}). \tag{7}$$

where $\mathcal{L}$ denotes the total number of layers. We denote the collection of node embeddings of *scene-graph* $\mathcal{G}_t$ after passing through $\mathcal{L}$ layers of MR-GCN as $\mathcal{X}_t^{prop}$.

We then utilize a graph pooling and readout operation to condense the set of node embeddings $\mathcal{X}_t^{prop}$ to a single, unified graph embedding $\mathbf{h}_{\mathcal{G}_t}$. Here we employ the *self-attention graph pooling* operation [22]. Specifically, in the pooling layer, nodes are pooled according to the scores predicted from a trainable graph convolutional networks (GCN) layer, denoted as **SCORE**, as

$$\alpha = \mathbf{SCORE}(\mathcal{X}_t^{prop}, \mathcal{A}_t) \quad \text{and} \quad \mathcal{P} = \text{top}_k(\alpha), \tag{8}$$

where α represents the attention coefficient output by the graph pooling layer for each node in $\mathcal{G}_t$, $\mathcal{P}$ represents the top k proportion of nodes ranked according to α, and k is usually a pre-defined pooling ratio (e.g., 0.25, 0.5, 0.75) as it is assumed that only some nodes in each scene-graph are most relevant to the risk assessment task. This pooling layer also helps to filter out noise and improve training convergence. The node embeddings and edge adjacency information after pooling by $\mathcal{X}_t^{pool}$ and $\mathcal{A}_t^{pool}$ are then calculated as

$$\mathcal{X}_t^{pool} = (\mathcal{X}_t^{prop} \odot \tanh(\alpha))_{\mathcal{P}}, \tag{9}$$

$$\mathcal{A}_t^{pool} = \mathcal{A}_t^{prop}{}_{(\mathcal{P},\mathcal{P})}. \tag{10}$$

where $\odot$ represents an element-wise multiplication, $(\cdot)_{\mathcal{P}}$ indicates the operation of extracting a subset of nodes based on $\mathcal{P}$, and $(\cdot)_{(\mathcal{P},\mathcal{P})}$ denotes the construction of the adjacency matrix between the nodes in this subset. The set of pooled nodes is then processed by a readout layer, compressing the node embeddings $\mathcal{X}_t^{pool}$ into a single graph embedding $\mathbf{h}_{\mathcal{G}_t}$,

$$\mathbf{h}_{\mathcal{G}_t} = \mathbf{READOUT}(\mathcal{X}_t^{pool}) \tag{11}$$

where the **READOUT** operation can be summation (*sum-pooling*), averaging (*mean-pooling*), or selecting the maximum of each feature dimension over the set of node embeddings (*max-pooling*).

3.3.2 Temporal Modeling

We employ a long short-term memory (LSTM) [13] network to convert the sequence of scene-graph embeddings to the spatial-temporal embedding $\mathcal{Z}$. For each timestamp t, the LSTM updates the hidden state p_t and cell state c_t as

$$p_t, c_t = \mathbf{LSTM}(\mathbf{h}_{\mathcal{G}_t}, c_{t-1}), \tag{12}$$

where $\mathbf{h}_{\mathcal{G}_t}$ is the final *scene-graph* embedding from timestamp t. After the LSTM processes all the scene-graph embeddings, a temporal readout operation is applied to the resulting output sequence to compute the final spatial-temporal embedding $\mathcal{Z}$ as

$$\mathcal{Z} = \mathbf{TEMPORAL\_READOUT}(p_1, p_2, ..., p_{\mathcal{T}}) \tag{13}$$

where the **TEMPORAL_READOUT** operation could be the extraction of only the last hidden state $p_{\mathcal{T}}$ (LSTM-last) or could be a temporal attention layer (LSTM-attn). Here we integrate an attention mechanism into the LSTM architecture, i.e., LSTM-attn, to boost model performance. By adding an attention layer between successive LSTM layers, the model can weigh the significance of each timestep in the sequence, allowing it to focus more on the most relevant parts of the data. Additionally, the LSTM-attn layer calculates a context vector by considering the entire hidden state sequence $\{p_1, p_2, ..., p_{\mathcal{T}}\}$ returned from the LSTM encoder layers, and can hence effectively enhance the model's ability to make accurate and contextually-aware predictions.

The last layer in our model generates an output risk classification $\hat{\mathcal{Y}}$ from the spatial-temporal embedding $\mathcal{Z}$ as

$$\hat{\mathcal{Y}} = Softmax(\mathbf{MLP}(\mathcal{Z})) \tag{14}$$

Since our model is implemented as a binary classifier, we use Cross-Entropy loss to train the model. i.e.

$$\arg\min \mathbf{CrossEntropyLoss}(\mathcal{Y}, \hat{\mathcal{Y}}) \tag{15}$$

4. Experimental Results

4.1. Experimental Setup

Platform: We conduct our experiments on a Linux server with an Intel Xeon E5 CPU and an NVIDIA TITAN Xp GPU for training and evaluating each model.

Dataset: Our evaluation utilized three different types of datasets: (i) simulated lane change scenarios of varying risk from Carla [12], denoted as *271-carla* and *1043-carla*; (ii) real-world, clear-weather safe driving in California Bay Area from Honda [31], denoted as *1361-honda*; and (iii) real-world crashes and dangerous road scenarios from dash-cam footage from the Detection of Traffic Anomaly Dataset [39], denoted as *620-dash*. We use the same dataset preparation steps as in [26]. The number of risky and non-risky scenes in each dataset is listed in Table 1. For the subjective risk assessment of each dataset, we use 70% of data for training and 30% of data for inference. For transfer learning experiments, we train with 70% of the training dataset and evaluate with 100% of the inference dataset, since the training and inference datasets are distinct.

Table 1. Detailed Breakdowns of Datasets

Dataset	Non-Risky Scenarios	Risky Scenarios	Non-Risky:Risky Ratio
271-carla	223	48	4.65:1
571-honda	475	99	4.80:1
620-dash	323	297	1.09:1
1043-carla	898	146	6.15:1
1361-honda	1207	154	7.84:1

Model Specification: Our proposed model consists of three main modules: graph extraction, spatial model, and temporal model. For graph extraction, we implement three variants of edge encoding methods for RS2G: (i) one-dimensional MLP, denoted as *RS2G(1D MLP)*, employs a node encoder of dimensions 15×15; given that the edge encoder processes features from two nodes simultaneously, its shape is 30×12. In particular, each node vector is of dimension 15, and hence the concatenation of two nodes is of dimension 30. We set 12 as the total number of relations; a trivial number of relations can impede the expressiveness of graph representations, while an overlarge number of relations can weaken graph representations by making the relations redundant. (ii) For two-dimensional MLP, denoted as *RS2G(2D MLP)*, both the node and edge encoders have one extra layer, with shapes of $15 \times 15 \times 15$ and $30 \times 30 \times 12$, respectively. (iii) For the Transformer variant, denoted as *RS2G(Transformer)*, the node encoder retains the $16 \times 32 \times 16$ shape as in the 2D MLP. The Transformer encoder itself is configured with `d_model` (indicating the expected number of features in the input) set to 32 and the number of layers is set to 8. Following this encoder, there's an accompanying MLP with dimensions 32×12 to finalize the feature transformation. Our spatial and temporal modeling follow the same structure of MR-GCN and LSTM as the downstream of [40] for fair comparisons. Specifically, for modeling spatial graph features, we utilize a 2-layer MR-GCN with self-attention graph pooling and *mean* readout. We then apply a 2-layer LSTM with temporal attention as the readout operation for our temporal model.

Baseline Models: We compare RS2G with (i) the SOTA rule-based graph extraction and learning approach [40], denoted as "Rule-Based" graph extraction in our experiments; and (ii) the SOTA DL-based approach utilizing the CNN+LSTM architecture [41], which we denote its graph extraction as "None" since this method does not use graphs.

Evaluation Metrics: As we model risk assessment as a binary classification task, we evaluate each model in terms of Accuracy, Matthews Correlation Coefficient (MCC) [6], and Area Under the ROC Curve (AUC) [3]. Accuracy in this case is the standard metric indicating the percentage of correctly classified scenes. AUC score is a typical metric for scoring classifiers across multiple decision boundaries; it ranges from 0.0 to 1.0 with higher performance indicating a more robust model. MCC score is considered a more reliable metric than accuracy for evaluating models on imbalanced datasets. Specifically, an MCC score of -1.0 represents an always wrong classifier, 1.0 represents an always correct classifier, and 0.0 represents a random classifier.

4.2. Subjective Risk Assessment

Table 2 demonstrates the performance of each model variant at subjective risk assessment on both synthetic datasets (*271-carla, 1043-carla*) and real-world driving datasets (*620-dash, 1361-honda*). Across all datasets, all the RS2G models (1D MLP, 2D MLP, and Transformer) demonstrate significantly higher accuracy than the SOTA DL-based model using no graph extractions and the SOTA rule-based graph extraction method, indicating that our data-driven graph extraction technique can effectively enhance graph representations and scenario understanding for AVs. Specifically, RS2G (Transformer) provides on average 21.35% higher accuracy than the SOTA DL-based approach and 4.47% higher accuracy than the SOTA rule-based graph extraction method. Additionally, our data-driven graph extraction method also provides considerably higher MCC and AUC scores, indicating notably better capabilities in distinguishing positive and negative samples. Furthermore, RS2G (Transformer) notably outperforms other edge encoding methods (1D MLP and 2D MLP) in most cases, proving that the attention mechanism of the Transformer can better capture complex relations among road users.

Overall, all the models provide lower learning quality, i.e., accuracy, MCC, and AUC, for real-world driving datasets than synthetic datasets. In particular, for a real-world imbalanced dataset with more crashes and risky scenarios, i.e., *620-dash*, the CNN-based model delivers seriously degraded accuracy and an MCC score worse than a random classifier (less than 0.0). However, all the RS2G models exhibit notably less performance degradation, indicating that our data-driven graph extraction technique can provide more effective performance in complex real-world scenarios than SOTA DL-based approaches and SOTA GL-based models using the rule-based graph extraction method.

4.3. Transfer Learning Evaluation

We evaluate the *Sim2Real* transfer learning capability of each model, i.e., the capability of generalizing knowledge gained from simulations to real-world scenarios. We first train each model on a simulation dataset (*271-carla* or *1043-carla*) and then evaluate the trained model on the real-world dataset (*620-dash*) consisting of considerably more instances of crash scenarios. Simulations in synthetic datasets only include lane-changing behaviors, while all driving maneuvers are presented in the real-world dataset; i.e., the visual context of the real-world dataset significantly differs from that in simulation environments. As demonstrated in Table 3, all the learning methods deliver degraded performance; however, RS2G models (2D MLP, Transformer) achieve notably higher accuracy, MCC, and AUC than the SOTA DL-based model and the SOTA GL-based method using rule-based graph extraction. Specifically, RS2G (Transformer) provides on average 14.68% higher accuracy than the SOTA DL-based model, 15.90 higher accuracy than the SOTA GL-based method using rule-based graph extraction, and 6.54% than RS2G (2D MLP).

Dataset	Graph Extraction	Accuracy	MCC	AUC
271-carla	None	73.17%	0.1887	0.8043
	Rule-Based	82.93%	0.5173	0.8098
	RS2G (1D MLP)	84.51%	0.2093	0.9338
	RS2G (2D MLP)	**86.59%**	**0.468**	**0.9578**
	RS2G (Transformer)	84.15%	0.402	0.9362
1043-carla	None	71.66%	0.1111	0.7173
	Rule-Based	91.43%	0.7217	0.971
	RS2G (1D MLP)	91.72%	0.6840	0.9643
	RS2G (2D MLP)	93.31%	0.7426	0.7949
	RS2G (Transformer)	**97.13%**	**0.8823**	**0.9686**
1361-honda	None	60.39%	0.0391	0.7110
	Rule-Based	86.31%	0.2445	0.9341
	RS2G (1D MLP)	87.04%	0.1626	0.9315
	RS2G (2D MLP)	89.00%	0.3029	0.9383
	RS2G (Transformer)	**89.98%**	**0.404**	**0.9495**
620-dash	None	48.92%	-0.1749	0.5256
	Rule-Based	67.20%	0.3428	0.6966
	RS2G (1D MLP)	68.82%	0.3967	0.7403
	RS2G (2D MLP)	**72.04%**	**0.4398**	**0.8047**
	RS2G (Transformer)	68.28%	0.3635	0.7354

Table 2. Performance of Subjective Risk Assessment for different graph extraction methods and datasets. "Rule-based" refers to the SOTA GL-based model based on rule-based graph extraction [40], and RS2G is our proposed approach. The downstream of these methods consists of an MR-GCN and an LSTM model, taking scene-graphs as input. "None" graph extraction refers to SOTA DL-based model [41] without using graphs, where the downstream task processes raw image data using a CNN and an LSTM model.

Dataset	Graph Extraction	Accuracy	MCC	AUC
271-carla to 620-dash	None	52.58%	0.0333	0.5126
	Rule-Based	48.22%	0.0238	0.4975
	RS2G(2D MLP)	57.25%	0.1398	0.5669
	RS2G(Transformer)	**64.68%**	**0.2957**	**0.6831**
1043-carla to 620-dash	None	49.03%	-0.0432	0.4999
	Rule-Based	50.96%	0.0021	0.5093
	RS2G(2D MLP)	60.65%	0.2089	0.6265
	RS2G (Transformer)	**66.29%**	**0.3293**	**0.6964**

Table 3. Transfer learning comparison between different the SOTA DL-based model [41], the SOTA GL-based model with rule-based graph extraction, RS2G (2D MLP) and RS2G (Transformer).

It is also noteworthy that the baseline models trained on *1043-carla* both deliver lower performance after transfer than the same models trained on *271-carla*, which contains notably fewer data samples than *1043-carla*. It is likely to be caused by their overfitting to the training domain, i.e., the simulation environments, thus degrading their performance in the test domain, i.e., real-world settings. In contrast, our data-driven RS2G, especially models utilizing the Transformer as the relation extractor, dynamically extract more expressive graph representations and therefore is less likely to encounter overfitting issues, hence providing better performance for Sim2Real transfer learning.

4.4. Ablation Study

Here we provide detailed ablation studies to further demonstrate the benefits of Kullback-Leibler(KL) divergence and results using different edge extraction thresholds. We present (i) the impact of each component of the downstream for GL-based models (ii) a similarity comparison between graphs extracted by the rule-based method and our data-driven approach in our supplementary material.

4.4.1 Analysis of Kullback-Leibler Divergence

Model		Accuracy	MCC	AUC
Graph Extraction	with KL			
RS2G(2MLP)	✗	62.42%	0.2455	0.6132
RS2G(2MLP)	✓	60.65%	0.2089	0.6265
RS2G(Transformer)	✗	64.35%	0.2897	0.6586
RS2G(Transformer)	✓	**66.29%**	**0.3293**	**0.6964**

Table 4. KL Analysis for RS2G (2D MLP) and RS2G (Transformer) in transfer learning from *1043-carla* to *620-dash*.

We present the effects of KL divergence on RS2G (2D MLP) and RS2G (Transformer) in Sim2Real transfer learning in Table 4. In particular, when the KL divergence is not involved in the model, RS2G (Transformer) outperforms RS2G (2D MLP) by 1.93%. However, after incorporating KL divergence into the model, the performance of RS2G (2D MLP) slightly drops whereas the performance of RS2G (Transformer) noticeably improves. We summarize the potential reason as follows: The scene-graphs generated from MLP tend to be simpler due to the simple MLP architecture which limits the model to capture sequential or relational intricacies in data. Thus, when KL regularization is introduced, it places an extra constraint on a graph that is already simple, pushing the model towards even more conservative graph representations. Consequently, the expressiveness of graphs becomes very limited, leading to less effective representations of intricate tasks or diverse conditions by RS2G (2D MLP). On the other hand, for RS2G (Transformer), given the power of the attention mechanism of the Transformer, its generated scene-graphs can be very comprehensive, sometimes at the risk of being overly complex. Incorporating KL divergence into the Transformer does not simply constrain the model; instead, it refines the Transformer attention, ensuring that the model captures the most representative relation of the adjacency matrix. Thus, combining the Transformer with KL divergence produces more balanced and expressive scene-graphs, and leads to higher accuracy, MCC, and AUC scores.

4.4.2 Graph Structure Comparison

We compare the structural differences between rule-based graphs and the data-driven graphs extracted by RS2G. In particular, We evaluate how the methods differ regarding graph sparsity and edge distribution, and correlate these metrics with risk assessment accuracy. We also identify how RS2G's edge extraction threshold (γ) affects the sparsity of generated graphs and the model's overall performance. Specifically, the threshold γ indicates the sigmoid score that $Encode_{edge}$ must overcome to add a given edge to the graph varying from 0 to 1, i.e., higher γ results in sparser graphs. Our results using Transformer graph extraction with different γ are shown in Table 5. Transformer extraction with various thresholds exhibits higher accuracy than rule-based graph extraction, and the best performance is achieved with $\gamma = 0.25$. Using $\gamma = 0.5$ and $\gamma = 0.75$ lowers the performance, possibly due to overfitting. On the other hand, using $\gamma = 0.75$ can reach a better performance than $\gamma = 0.5$ may due to some seeming irrelevant nodes could actually provide useful information for the model.

Graph Ext.	Acc.	Avg. Deg.	Avg. Edges	σ Edges
Rule-Based	95.86%	3.84	16.50	10.51
RS2G ($\gamma = 0.25$)	**97.13%**	**37.11**	**298.98**	**264.36**
RS2G ($\gamma = 0.5$)	94.59%	23.98	193.21	171.22
RS2G ($\gamma = 0.75$)	95.54%	10.88	87.68	78.00

Table 5. Comparison of graph structure metrics between rule-based graph extraction [40] and RS2G (Transformer). γ represents the edge extraction decision threshold.

5. Conclusion

In this paper, we propose RS2G, a novel road scene understanding framework based on a innovative data-driven graph extraction and modeling approach, dynamically capturing complex relations among road users. RS2G learns to specialize relations with data-driven vectors, thereby providing more expressive graph representations of road scenes. We also leverage the powerful attention mechanism of the Transformer and the variational autoencoder to further enhance RS2G's capability to model relations and transfer knowledge from training domains to real-world scenarios. Our evaluation shows that RS2G significantly outperforms the SOTA DL-based model and the SOTA rule-based graph extraction method in both subjective risk assessment and Sim2Real transfer learning.

References

[1] Shivam Akhauri, Laura Zheng, Tom Goldstein, and Ming Lin. Improving generalization of transfer learning across domains using spatio-temporal features in autonomous driving. *arXiv preprint arXiv:2103.08116*, 2021.

[2] Hermann Blum, Paul-Edouard Sarlin, Juan Nieto, Roland Siegwart, and Cesar Cadena. Fishyscapes: A benchmark for safe semantic segmentation in autonomous driving. In *proceedings of the IEEE/CVF international conference on computer vision workshops*, pages 0–0, 2019.

[3] Andrew P Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern recognition*, 30(7):1145–1159, 1997.

[4] Sergio Casas, Cole Gulino, Renjie Liao, and Raquel Urtasun. Spagnn: Spatially-aware graph neural networks for relational behavior forecasting from sensor data. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9491–9497. IEEE, 2020.

[5] Bi-ke Chen, Chen Gong, and Jian Yang. Importance-aware semantic segmentation for autonomous driving system. In *IJCAI*, pages 1504–1510, 2017.

[6] Davide Chicco and Giuseppe Jurman. The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC genomics*, 21(1):6, 2020.

[7] Chiho Choi, Joon Hee Choi, Jiachen Li, and Srikanth Malla. Shared cross-modal trajectory prediction for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 244–253, 2021.

[8] Jiwoong Choi, Dayoung Chun, Hyun Kim, and Hyuk-Jae Lee. Gaussian yolov3: An accurate and fast object detector using localization uncertainty for autonomous driving. In *Proceedings of the IEEE/CVF International conference on computer vision*, pages 502–511, 2019.

[9] Felipe Codevilla, Eder Santana, Antonio M López, and Adrien Gaidon. Exploring the limitations of behavior cloning for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9329–9338, 2019.

[10] Mohammad Zalbagi Darestani, Akshay S Chaudhari, and Reinhard Heckel. Measuring robustness in deep learning based compressive sensing. In *International Conference on Machine Learning*, pages 2433–2444. PMLR, 2021.

[11] Nemanja Djuric, Vladan Radosavljevic, Henggang Cui, Thi Nguyen, Fang-Chieh Chou, Tsung-Han Lin, Nitin Singh, and Jeff Schneider. Uncertainty-aware short-term motion prediction of traffic actors for autonomous driving. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2095–2104, 2020.

[12] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. *arXiv preprint arXiv:1711.03938*, 2017.

[13] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[14] Sebastian Höfer, Kostas Bekris, Ankur Handa, Juan Camilo Gamboa, Melissa Mozifian, Florian Golemo, Chris Atkeson, Dieter Fox, Ken Goldberg, John Leonard, et al. Sim2real in robotics and automation: Applications and challenges. *IEEE transactions on automation science and engineering*, 18(2):398–400, 2021.

[15] Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, et al. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17853–17862, 2023.

[16] Xinyu Huang, Xinjing Cheng, Qichuan Geng, Binbin Cao, Dingfu Zhou, Peng Wang, Yuanqing Lin, and Ruigang Yang. The apolloscape dataset for autonomous driving. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 954–960, 2018.

[17] Jiman Kim and Chanjong Park. End-to-end ego lane estimation based on sequential transfer learning for self-driving cars. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 30–38, 2017.

[18] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[19] Pawit Kochakarn, Daniele De Martini, Daniel Omeiza, and Lars Kunze. Explainable action prediction through self-supervision on scene graphs. *arXiv preprint arXiv:2302.03477*, 2023.

[20] Nick Lamm, Shashank Jaiprakash, Malavika Srikanth, and Iddo Drori. Vehicle trajectory prediction by transfer learning of semi-supervised models. *arXiv preprint arXiv:2007.06781*, 2020.

[21] Pat Langley, Ben Meadows, Mohan Sridharan, and Dongkyu Choi. Explainable agency for intelligent autonomous systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, pages 4762–4763, 2017.

[22] Junhyun Lee, Inyeop Lee, and Jaewoo Kang. Self-attention graph pooling. *arXiv preprint arXiv:1904.08082*, 2019.

[23] Chengxi Li, Yue Meng, Stanley H Chan, and Yi-Ting Chen. Learning 3d-aware egocentric spatial-temporal interaction via graph convolutional networks. *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8418–8424, 2020.

[24] Peixuan Li, Shun Su, and Huaici Zhao. Rts3d: Real-time stereo 3d detection from 4d feature-consistency embedding space for autonomous driving. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 1930–1939, 2021.

[25] Xiwen Liang, Minzhe Niu, Jianhua Han, Hang Xu, Chunjing Xu, and Xiaodan Liang. Visual exemplar driven task-prompting for unified perception in autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9611–9621, 2023.

[26] Arnav Vaibhav Malawade, Shih-Yuan Yu, Brandon Hsu, Deepan Muthirayan, Pramod P Khargonekar, and Mohammad Abdullah Al Faruque. Spatiotemporal scene-graph embedding for autonomous vehicle collision prediction. *IEEE Internet of Things Journal*, 9(12):9379–9388, 2022.

[27] Ana I Maqueda, Antonio Loquercio, Guillermo Gallego, Narciso García, and Davide Scaramuzza. Event-based vision

meets deep learning on steering prediction for self-driving cars. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5419–5427, 2018.

[28] Sravan Mylavarapu, Mahtab Sandhu, Priyesh Vijayan, K Madhava Krishna, Balaraman Ravindran, and Anoop Namboodiri. Towards accurate vehicle behaviour classification with multi-relational graph convolutional networks. *arXiv preprint arXiv:2002.00786*, 2020.

[29] Sravan Mylavarapu, Mahtab Sandhu, Priyesh Vijayan, K Madhava Krishna, Balaraman Ravindran, and Anoop Namboodiri. Understanding dynamic scenes using graph convolution networks. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8279–8286. IEEE, 2020.

[30] Shuteng Niu, Yongxin Liu, Jian Wang, and Houbing Song. A decade survey of transfer learning (2010–2020). *IEEE Transactions on Artificial Intelligence*, 1(2):151–166, 2020.

[31] Vasili Ramanishka, Yi-Ting Chen, Teruhisa Misu, and Kate Saenko. Toward driving scene understanding: A dataset for learning driver behavior and causal reasoning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7699–7707, 2018.

[32] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *European Conference on Computer Vision*, pages 683–700. Springer, 2020.

[33] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. *European Semantic Web Conference*, pages 593–607, 2018.

[34] Hao Shao, Letian Wang, Ruobing Chen, Hongsheng Li, and Yu Liu. Safety-enhanced autonomous driving using interpretable sensor fusion transformer. In *Conference on Robot Learning*, pages 726–737. PMLR, 2023.

[35] Mark Strickland, Georgios Fainekos, and Heni Ben Amor. Deep predictive models for collision risk assessment in autonomous driving. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4685–4692. IEEE, 2018.

[36] Yafu Tian, Alexander Carballo, Ruifeng Li, and Kazuya Takeda. Road scene graph: A semantic graph-based scene representation dataset for intelligent vehicles. *arXiv preprint arXiv:2011.13588*, 2020.

[37] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017.

[38] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

[39] Yu Yao, Xizi Wang, Mingze Xu, Zelin Pu, Ella Atkins, and David Crandall. When, where, and what? a new dataset for anomaly detection in driving videos. *arXiv preprint arXiv:2004.03044*, 2020.

[40] Shih-Yuan Yu, Arnav Vaibhav Malawade, Deepan Muthirayan, Pramod P Khargonekar, and Mohammad Abdullah Al Faruque. Scene-graph augmented data-driven risk assessment of autonomous vehicle decisions. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[41] Ekim Yurtsever, Yongkang Liu, Jacob Lambert, Chiyomi Miyajima, Eijiro Takeuchi, Kazuya Takeda, and John HL Hansen. Risky action recognition in lane change video clips using deep spatiotemporal networks with segmentation mask transfer. *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 3100–3107, 2019.

[42] Cheng Zhang, Kun Zhang, and Yingzhen Li. A causal view on robustness of neural networks. *Advances in Neural Information Processing Systems*, 33:289–301, 2020.

[43] Ethan Zhang, Sion Pizzi, and Neda Masoud. A learning-based method for predicting heterogeneous traffic agent trajectories: implications for transfer learning. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 1853–1858. IEEE, 2021.

[44] Qingzhao Zhang, Shengtuo Hu, Jiachen Sun, Qi Alfred Chen, and Z Morley Mao. On adversarial robustness of trajectory prediction for autonomous vehicles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15159–15168, 2022.

[45] Zijian Zhu, Yichi Zhang, Hai Chen, Yinpeng Dong, Shu Zhao, Wenbo Ding, Jiachen Zhong, and Shibao Zheng. Understanding the robustness of 3d object detection with bird's-eye-view representations in autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21600–21610, 2023.

[46] Shlomo Zilberstein. Building strong semi-autonomous systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015.