# nuScenes Knowledge Graph - A comprehensive semantic representation of traffic scenes for trajectory prediction

Leon Mlodzian[1,2], Zhigang Sun[2], Hendrik Berkemeyer[3,5], Sebastian Monka[2,3], Zixu Wang[2,4], Stefan Dietze[1], Lavdim Halilaj[2,3], and Juergen Luettin[2,3]

[1]Heinrich Heine University Düsseldorf
[2]Bosch Center for Artificial Intelligence
[3]Robert Bosch GmbH
[4]Technical University of Munich
[5]University of Osnabrück

leon.mlodzian@proton.me, zhigang.sun3@cn.bosch.com, {hendrik.berkemeyer, sebastian.monka, lavdim.halilaj, juergen.luettin}@de.bosch.com, zixu.wang@tum.de, stefan.dietze@hhu.de

## Abstract

*Trajectory prediction in traffic scenes involves accurately forecasting the behaviour of surrounding vehicles. To achieve this objective it is crucial to consider contextual information, including the driving path of vehicles, road topology, lane dividers, and traffic rules. Although studies demonstrated the potential of leveraging heterogeneous context for improving trajectory prediction, state-of-the-art deep learning approaches still rely on a limited subset of this information. This is mainly due to the limited availability of comprehensive representations. This paper presents an approach that utilizes knowledge graphs to model the diverse entities and their semantic connections within traffic scenes. Further, we present nuScenes Knowledge Graph (nSKG), a knowledge graph for the nuScenes dataset, that models explicitly all scene participants and road elements, as well as their semantic and spatial relationships. To facilitate the usage of the nSKG via graph neural networks for trajectory prediction, we provide the data in a format, ready-to-use by the $PyG$ library. All artefacts can be found here: https://tinyurl.com/5t2vv9yu.*

## 1. Introduction

Traffic trajectory prediction is a crucial component of autonomous driving, as it enables the autonomous vehicle to anticipate the movement of other traffic participants and avoid dangerous situations that could lead to collisions. Deep learning approaches have proven to be very successful when applied to this task. A key driver behind the significant progress in deep learning is the availability of easily accessible datasets that have been compiled over the years. For instance, MNIST [41], COCO [47] and ImageNet [20] were crucial for progress in computer vision, GLUE [69] and SQuAD [58] for natural language understanding and MuJoCo [65] and OpenAI Gym [10] for reinforcement learning. The same is valid for trajectory prediction and includes datasets, such as Argoverse [15], Apolloscape [51], Interaction [72], and nuScenes [14]. However, there are two shortcomings of current approaches in trajectory prediction: (1) shortcomings of deep learning and (2) shortcomings in rich scene representation. We will describe them in more detail in the following sections.

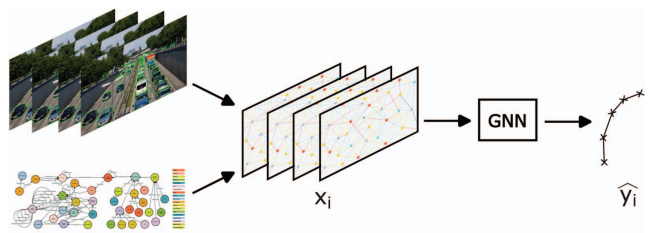


Figure 1. We model traffic scenes (top left) by applying a rigorous ontology (bottom left) to them, producing rich, temporal, heterogeneous graphs. We provide a large graph regression dataset of $(x_i, y_i)$ pairs for training GNNs on the designed representation. Partial image credits: rawpixel.com on Freepik.

**Shortcomings of deep learning** have been the subject of several investigations over the last years. Specifically, their lack of robustness [64, 52], explainability [37, 33, 48] as well as the inability to generalise to new domains [73, 7] [6]. One possible explanation for these limitations is that they operate purely on a sub-symbolic [53] and statistical basis, thus only learning correlations between input features and target variable, rather than attaining a causal, structured comprehension of a task [60, 4, 61]. Furthermore, for real-world applications of autonomous systems it is vital to consider safety aspects. ISO 26262 [35], the international standard for the functional safety of road vehicles, needs to be satisfied. Challenges in validation when using machine learning methods have been described in [12].

Studies suggest that humans do not reason at the pixel level but use attention and expectation at the object level [62, 19] to do predictive coding [16, 13]. Moreover, we possess inherent prior knowledge, such as intuitive physics and common sense [57] that we use in tasks like trajectory prediction. This high-level, structured information (knowledge) is typically missing when deep learning models are trained in end-to-end scenarios from raw data. We address this shortcoming by providing a semantic representation of the driving scene that can be exploited by deep learning based approaches.

**Shortcomings in rich scene representation** describes the situation that trajectory prediction datasets described above, lack in rich scene representation. Especially map and scene context information is rarely included. nuScenes is a unique dataset for trajectory prediction that stands out due to its comprehensive map information. However, the trajectory prediction community has not fully exploited the detailed heterogeneous map data because it is not provided in an easy to use data representation. Knowledge graphs [33], on the other hand, are well suited to represent and reason over structured and high-level information.

**In this work**, we provide a solution to address both shortcomings, deep learning and rich scene representation. We leverage the power of knowledge graphs to provide a comprehensive representation of the driving scene, forming a graph-based, symbolic representation at an intermediate level of abstraction. We implement our approach for the nuScenes dataset and provide the nuScenes Knowledge Graph (nSKG), a comprehensive, semantic representation of driving scenes. nSKG utilizes subject-predicate-object triples to structure high-level information. It is based on a rigorous ontology to model concepts such as agents (traffic participants) and map, their hierarchies and relationships. It is a rich representation of carpark areas, walkways, pedestrian crossings, lane geometry, and other map elements as well as traffic participants, their trajectories and semantic relations, including spatio-temporal relations between entities. Furthermore, we extract a nuScenes trajectory prediction graph dataset (nSTP) to alleviate data engineering efforts for neural network designers. It includes the wealth of relevant information from the knowledge graph and thus forms a new scene graph dataset that enables training graph neural networks (GNNs) on our rich scene representation. Both resources together enable symbolic (nSKG) and sub-symbolic (nSTP) methods to be explored for trajectory prediction with a wealth of structured information, previously only available in unstructured form. Neuro-symbolic AI has been dubbed the third wave of AI [18] based on the conjecture that the fusion of symbolic and sub-symbolic methods could relieve intelligent systems from the disadvantages of each. This could help to obtain explainable models that meet the safety requirements of autonomous vehicles.

The **main contributions** are:

- A comprehensive agent and map ontology that models driving scenes in detail.
- nSKG, a knowledge graph generated for the nuScenes dataset, based on the defined ontology.
- nSTP, a ready-to-use scene graph dataset for training GNNs for trajectory prediction.

The next section summarises the related work. Section 3 presents our ontology design for modeling traffic scenes as well as the generation of the nuScenes Knowledge Graph. Section 4 describes the construction of our readily usable graph dataset for trajectory prediction. Section 5 states limitations of our work and conclusions follow in the final section.

## 2. Related work

### 2.1. Trajectory prediction

One of the first set of neural networks applied to trajectory prediction were raster-based approaches [17, 22, 55, 9]. These approaches encode the traffic scene into birds-eye-view images with a number of channels. The channels are used to represent the various kinds of structures and agents in a scene. On top of these raster-representations, convolutional neural networks [40] are applied to learn a representation of the map and agents. Drawback of these models is that they do not have access to high-level information and need to learn from raw pixels.

The next generation of trajectory prediction techniques used a more natural and powerful data representation approach: graphs [45, 42, 26, 46, 43, 68, 29, 49]. These are higher-level data representations that do not require networks to learn from low-level pixels, which yields performance improvements. State-of-the-art approaches use these graphs for data representation. The various methods model scenes at different levels of abstraction. Methods like VectorNet [26] use fine representations, where nodes are simply coordinates and in combination with edges between them, they represent map structure borders or vehicle trajectories. On the other hand, very recent approaches [29, 75] use high-level representations, where single nodes represent whole entities, like vehicles or lanes. For such high-level representations, heterogeneous graphs are employed to capture the different types of nodes and edges that arise.

| | Lane center | Lane width | Lane border | Border type | Stop area | Traffic light | Traffic signs | Cross-ing | Walk-way | Car park | Agent relations |
|---|---|---|---|---|---|---|---|---|---|---|---|
| VectorNet [26] | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| LaneGCN [46] | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| Holistic [29] | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| Relation [75] | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| PGP [21] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ |
| HDGT [38] | ✓ | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ |
| LAformer [49] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ |
| **Ours** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

Table 1. Comparison of information included in popular and state-of-the-art trajectory prediction approaches. Raster-based methods [17, 55] were not included due to their inferior performance and non-explicit information structure.

Graph neural networks are the standard method for learning on graphs. Heterogeneous graphs are either used in conjunction with a heterogeneous graph neural network [29], or the types of nodes and edges is categorically encoded into a feature and then processed by a standard (homogeneous) graph neural network [26, 75].

Traffic representations designed for trajectory prediction have become more structured and high-level over time. From rasters to simple graphs, from simple graphs to heterogeneous graphs and this work takes another step, namely knowledge graphs.

### 2.2. Map representation

Recently, rich map context has received increased attention and is considered to be an important cornerstone in reaching further improvements in trajectory prediction [46]. It is an open research question how the complex and rich road topology with lanes, walkways, car parks, traffic signs, pedestrian crossings and traffic lights is best represented and how much this aids trajectory prediction. No previous work has been found that uses available map information comprehensively (see table 1). Although being widely considered important [46, 26, 21], the large majority of map information has so far been ignored, possibly due to the high engineering effort to obtain an easily usable data representation. State-of-the-art results in trajectory prediction were reached by [21] which includes lane center points, pedestrian crossings and stop area information. We hypothesise that results can be improved by representing more diverse road elements and semantic relational information.

Looking beyond trajectory prediction, there are other branches of automated driving interested in how maps can be represented. A recent survey on knowledge graphs for automated driving [50] contains a comprehensive list of available ontologies, only one of which has a focus on the map [63]. It is a small ontology with only seven concepts that explores the feasibility of using ontologies for driver assistance functions. The map structures that it models are lanes, traffic signs and road pieces. On the other extreme, [71] uses description logic reasoning to recognise the criticality of driving situations. Things like whether the road is wet or sandy, what a traffic light's color is and much else is considered. The ontology is very complex with a large number of concepts, the large majority of which cannot be generated from trajectory prediction datasets since such information is neither directly included nor derivable.

### 2.3. Trajectory representation

For modelling the trajectories of participants, relevant schemas exist. [31] and [34] both propose an ontology for modelling agent data. The main difference between them is that the former is agent-centric whereas the latter is trajectory-centric. The agent-centric model includes a notion of agents at certain timesteps. It only models agents as time-independent. In trajectory prediction, one cares about how properties of agents like speed and orientation evolve, making an agent-centric model suitable.

### 2.4. Ontologies in autonomous driving

Some well-known general ontologies that contain concepts related to autonomous driving (AD) include SOSA [36], DBpedia [5] and Schema.org [30]. A survey that compares and contrasts available ontologies in AD can be found in [50]. More specific works include [27, 66] that intend to create a shared vocabulary across AD applications. There exist ontologies that model vehicles [74] and sensors [39]. Human driver modelling has received attention in [32, 24] and particularly in [59] where demographic and behavioural aspects are considered. A context model for automated vehicles is presented in [67]. This models some of the aspects we are interested in, but many relevant factors for trajectory prediction are not modelled. Lastly, there are standardisation efforts for modelling the central concepts in AD with ontologies, for example ASAM OpenX [3] and ASAM OpenScenario [2]. So far, these ontologies have been hardly used due the lack of available data. Here, we design an ontology to be applied for AD that represents data that is typically expected to be available in future AD systems. As first implementation, we choose to use the nuScenes dataset, one of the most widely used datasets in autonomous driving

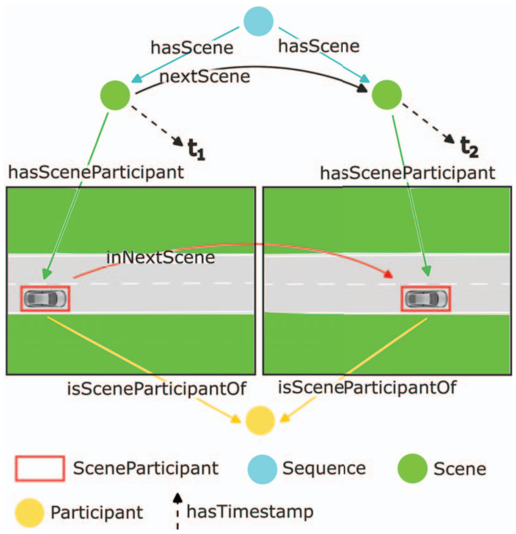


Figure 2. Model of the temporal nature of traffic scenarios applied to a single car travelling along a lane.

that contains rich map information and that was recorded by a stat-of-the-art sensor suite of Lidar, Radar cameras, IMU and GPS sensor.

# 3. Ontology and knowledge graph generation

To describe the design of the ontology and the generation of the knowledge graph, we first introduce a concept, then a $\mathcal{SROIQ}(\mathcal{D})$ (on which OWL 2 [54] is based) description logic formalisation is given, and finally the knowledge graph instance generation from nuScenes is explained. The ontology is a generic traffic scene model that can be applied to other datasets or extended to represent new pieces of scene information in future.

## 3.1. Temporal representation

**Sequence, Scene.** We reuse the concepts *Sequence* and *Scene* from [31] to divide a driving situation. A *Scene* refers to a single moment in a traffic situation. *hasTimestamp* is an integer data property with a unix timestamp defining the moment in time. A *Sequence* is an ordered collection of *Scene*s. A *Sequence* can be thought of as a video where its frames are *Scene*s. Since the order of *Scene*s is inherent in them, object properties *hasNextScene* and *hasPreviousScene* are defined to link consecutive *Scene*s.

$$\begin{aligned} Scene \equiv\ & \exists hasNextScene.Scene \\ & \cup \exists hasPreviousScene.Scene \end{aligned} \tag{1}$$

$$Sequence \equiv \exists hasScene.Scene \tag{2}$$

*Sequence* and *Scene* instances are generated from the SCENE and SAMPLE nuScenes records, respectively.

**Trip, Location.** *Sequence*s refer to specific trajectory prediction situations. During recording of motion data the ego-vehicle might travel for hours and record several *Sequences*. A *Trip* is such a recording session and each of its entities points to several *Sequence*s. Each *Trip* is taken in a particular region of interest, a *Location*, related to it via *hasLocation*. *hasRightHandTraffic* is a boolean property to describe the driving direction at a *Location*.

$$Trip \equiv \exists hasSequence.Sequence \tag{3}$$

$$Location \equiv \exists hasLocation^{-1}.Trip \tag{4}$$

*Trip* instances are generated from nuScenes LOG records and a *Location* instance is manually created for each of the four maps.

## 3.2. Participant representation

**Participant, SceneParticipant.** The *Participant* concept represents a traffic agent present in one or multiple *Scene*s. The various types of participants are modelled as subclasses of the *Participant* concept. There are in total 23 different ones. Examples are cars, adults, children, police officers, ambulances, bicycles and so on. In [31], *Participant*s refer to an entity at a certain timestep. A new relation *inNextScene* was introduced to be able to link entities across time. Further, the concept *SceneParticipant* was introduced as a notion of an agent at a certain timestep and the meaning of *Participant* was changed to represent an agent generally, independent of time. This avoids having to store time-independent information, e.g. sizes of agents, redundantly. The semantic relationship between *SceneParticipant*s is modelled as in [75], where agents may follow one another (longitudinal), potentially intersect (intersecting) or be parallel (lateral) to one another (see figure 3).

$$\begin{aligned} & SceneParticipant \equiv \\ & \quad \exists hasSceneParticipant^{-1}.Scene \cap \\ & \quad \exists isSceneParticipantOf.Participant \end{aligned} \tag{5}$$

$$\begin{aligned} & Participant \equiv \\ & \quad \exists isSceneParticipantOf^{-1}.SceneParticipant \end{aligned} \tag{6}$$

*SceneParticipant* instances are generated from SAMPLE_ANNOTATION and EGO_POSE records. The EGO_POSE records are needed such that the ego-vehicle can be included as a *SceneParticipant*. This is a novelty in our data representation. Previous work has ignored the effect of the ego-vehicle on the target vehicle's motion. Our data analysis of the nuScenes dataset shows that ego and target can be up to 2m close in a significant number of cases. We therefore expect the ego-vehicle to have an influence on the target vehicle's behaviour.

### 3.3. Lane representation

**Lane, LaneConnector.** The central component of road traffic infrastructure is the *Lane*. This is defined as a non-overlapping stretch of road surface, typically confined by lane borders, where only one driving direction is allowed. This is a physical lane formalisation as opposed to a logical one, where lanes go across junctions and can overlap [56]. To keep the logical connectivity information with the physical definition, one needs *LaneConnector*s, which have the functional properties *hasIncomingLane* and *hasOutgoingLane* pointing to a *Lane* each.

$$\begin{aligned} Lane \equiv\ & \exists hasNextLane.Lane \\ & \cup \exists hasPreviousLane.Lane \\ & \cup \exists hasLeftLane.Lane \\ & \cup \exists hasRightLane.Lane \end{aligned} \tag{7}$$

$$\begin{aligned} LaneConnector \equiv\ & \exists hasIncomingLane.Lane \\ & \cap \exists hasOutgoingLane.Lane \end{aligned} \tag{8}$$

*Lane* and *LaneConnector* instances are generated from LANE and LANE_CONNECTOR records, respectively.

**LaneSnippet, switchVia.** Lane borders are another crucial element determining how cars travel. Different lane divider types exist, such as solid lines and dashed lines. A *LaneSnippet* is defined as a piece of a lane that has a single border type on each its left and its right side. This allows the introduction of a *switchVia* property for every type of border, i.e. *switchViaDoubleDashed*, *switchViaSingleSolid*, etc. Neighbouring snippets that have a , say, single solid border between them, get related to one another via *switchViaSingleSolid*, representing that a single solid border would have to be crossed to switch from one to the other. Switches via borders that are illegal are kept in the model because cars may sometimes break traffic rules and overtake across a solid border, for example. *hasNextLaneSnippet* points from one snippet to the immediately following one and *hasLaneSnippet* keeps them connected to their parent *Lane*. Further, since experimental evidence [21] has shown that it is important for trajectory prediction performance to keep snippets short, they are further divided if they exceed 20 meters in length. *snippetHasLength* keeps a record of how long a particular lane snippet is.

$$\begin{aligned} LaneSnippet \equiv\ & \exists switchViaDoubleDashed.LaneSnippet \\ & \cup \exists switchViaSingleSolid.LaneSnippet \\ & \cup \dots \end{aligned} \tag{9}$$

*LaneSnippet* instances were computed from LANE records. The border types (solid line, dashed line, etc.) on each side of a lane were tracked and split into sections that have non-changing border types on either side. Sections were divided, if necessary, to satisfy the 20m length bound. This produced *LaneSnippet* instances with constant border types on either side. *switchVia* edges were placed between neighbouring *LaneSnippet* instances.

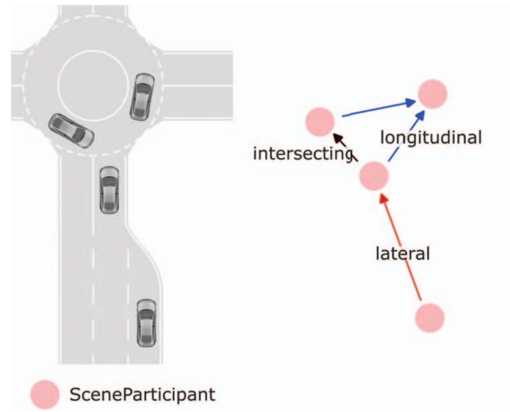

Figure 3. Semantic relationship model between agents.

**LaneSlice, OrderedPose.** To represent the centerlines (where cars typically drive) of lanes and lane connectors, a sequence of *Pose*s is used. A *Pose* consists of a position and an orientation. The orientation here denotes the orientation of the lane, i.e. the traffic direction, at a certain position. An *OrderedPose* is a subclass of *Pose* that also has the *hasNextPose* property. This is used to order them, defining the typical trajectory along a *LaneConnector* via the *connectorHasPose* relation to all its ordered poses. A *Pose*'s position, is modelled with *sf:Point* as are the agent positions, and its orientation with data property *poseHasOrientation*, represented as the angle between the positive x-axis and the direction facing (yaw). Contrary to lane connectors, the lane model needs to satisfy competency questions about width, too. The natural naming *LaneSlice* is chosen to represent the combination of center pose and lane width. *hasNextLaneSlice* keeps them ordered by connecting consecutive slices, *hasLaneSlice* points from parent lane to its slices and *laneSliceHasWidth* is the data property the name suggests.

$$\begin{aligned} LaneSlice \equiv\ & \exists laneHasSlice^{-1}.Lane \\ & \cap \exists laneSliceHasWidth.\mathbb{R} \end{aligned} \tag{10}$$

$$\begin{aligned} OrderedPose \equiv\ & \exists connectorHasPose^{-1}. \\ & LaneConnector \end{aligned} \tag{11}$$

$$OrderedPose \sqsubseteq Pose \tag{12}$$

*OrderedPose* instances were generated from the hand-annotated arclines from nuScenes at a resolution of 2m with the aid of the nuscenes-devkit. *LaneSlice* instances additionally represent lane width. A given center point, for which width is to be computed, is projected to both left

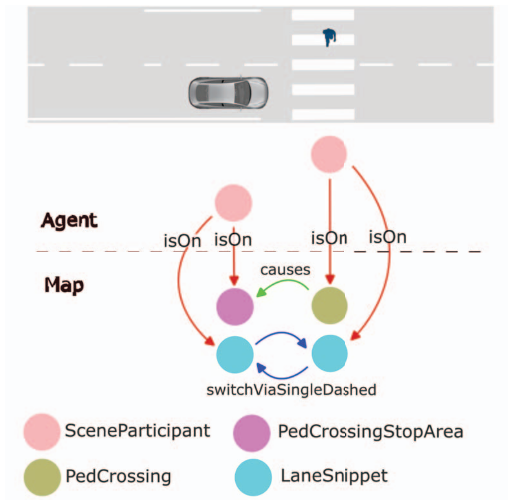


Figure 4. Example of how *isOn* models the spatial relation between agents and map elements. In addition, the given scenario illustrates stop areas and lane snippets.

and right borders. The projected points are those points on the borders that have the smallest Euclidean distance to the given center point. The width is given by the distance between the projected points.

### 3.4. Road infrastructure representation

**StopArea.** Stop areas are a very important concept for trajectory prediction because they, by definition, are the regions where cars tend to come to a halt. Several reasons exist for such regions and each is modelled as a subclass of the parent class *StopArea*. These include stop signs, yield signs, oncoming traffic when wanting to make a left turn, pedestrian crossings and traffic lights. *causesStopAt* link the causing entity to their associated *StopArea*.

$$\begin{aligned} StopArea \equiv\ & PedCrossingStopArea \\ & \cup\ TrafficLightStopArea \cup YieldStopArea \\ & \cup\ StopSignArea \cup TurnStopArea \end{aligned} \tag{13}$$

*StopArea* instances were generated from nuScenes STOP_LINE records.

**TrafficLight.** *hasTrafficLightType* differentiates horizontally and vertially stacked traffic lights. In addition, the lights are at a certain position and face a certain way, which is represented via *trafficLightHasPose* pointing to a particular *Pose* instance. The dynamic state of traffic lights (light color) is not modelled because this information is not available in the nuScenes dataset.

$$\begin{aligned} TrafficLight \equiv\ & \exists trafficLightHasPose.Pose \\ & \cap \exists hasTrafficLightType.\{H, V\} \end{aligned} \tag{14}$$

*TrafficLight* instances were generated from nuScenes TRAFFIC_LIGHT records.

**PedCrossing.** This is where pedestrians can legally cross the road. The two walkways connected via a crossing are represented with the *connectsWalkways* relation.

$$PedCrossing \equiv \leq 2 connectsWalkways.Walkway \tag{15}$$

Inspections of crossings and walkways in the nuScenes dataset showed that they often don't touch, but are always in close proximity. As a heuristic, walkways within a 5 m distance of a crossing were considered. Our algorithm chooses the two walkways with minimal distances. To check implementation correctness, a subset of generated triples were visualised and verified.

**Walkway, CarparkArea.** Walkways are modelled with a concept of the same name. *CarparkArea* is any area where cars can park, be that on an actual carpark or by the side of a road. To represent proximity between neighbouring parts of the road explicitly, *isNextTo* exists.

$$Walkway \equiv \exists walkwayIsNextTo.Lane \tag{16}$$

$$CarparkArea \equiv \exists carparkIsNextTo.Lane \tag{17}$$

The *isNextTo* relation between walkways, lanes and carparks is generated for those pairs of entities that are within 4 m distance. This heuristic threshold was chosen after visualising several lanes, carparks and walkways and their proximities. This way an explicit spatial relation is established between neighbouring pavement surfaces.

**RoadBlock.** Road blocks group adjacent lanes that go in the same direction. A *hasNextRoadBlock* edge exists from one block to another, if they contain lanes that follow one another. Road block connectivity therefore models any potential future region a car can go. Further, a *hasOpposingRoadBlock* relation is introduced. It exists between two road blocks if they are parallel to each other on the same road, carrying traffic in opposite directions. This extends the spatial connectivity in the graph, making spatial relations explicit that humans see intuitively.

$$RoadBlock \equiv \exists hasNextRoadBlock.RoadBlock \tag{18}$$

*RoadBlock* instances were computed by grouping neighbouring *Lane*s and the connectivity between road blocks was dictated by the lane connectivity. Instances could not be generated from nuScenes ROAD_BLOCK records because they contained malformed shapes on two of the four maps, as was raised in a GitHub issue and confirmed by Motional [1].

**Intersection.** This is where multiple lanes cross. The typical paths traversed across intersections are defined by lane connectors. A lane going into the intersection is connected to the outgoing lanes that may be travelled to legally. *isConnectorOnRoadSegment* relates intersections to the lane connectors on them.

$$\begin{aligned} Intersection \equiv \exists isConnectorOnRoadSegment^{-1}. \\ LaneConnector \end{aligned} \tag{19}$$

*Intersection* instances were generated from ROAD_SEGMENT records. The explicit spatial link between them and lane connectors was computed by checking whether a lane connector overlaps with an intersection.

**hasShape.** To model the precise positions, shapes and sizes of all map elements described above, *hasShape* relations are introduced for each. Each shape is represented with a subclass of the GeoSPARQL Simple Features (prefix *sf*) ontology concept *sf:Geometry*. It includes *sf:Polygon*, for example, which is used to model polygonal structures like walkways, lanes or intersections. Data properties of geometries store their precise shapes in nuScenes (x, y) coordinates, but also GPS coordinates, representing the real location on Earth. This enables fusion with other geographic data sources and geospatial analysis.

**isOn, AreaElement.** To create a connection between agents and the map, the *isOn* relation is introduced. *AreaElement* is defined as a superclass for all map elements that occupy an area, i.e. have a *sf:Polygon* geometry. *isOn* links a *SceneParticipant* to the map object it's currently on.

$$\begin{aligned} &AreaElement \equiv Walkway \cup CarparkArea \\ &\quad \cup Lane \cup LaneSnippet \cup RoadBlock \\ &\quad \cup StopArea \cup PedCrossing \cup Intersection \end{aligned} \tag{20}$$

The entire nSKG contains 56 million triples.

# 4. nuScenes Trajectory Prediction dataset

Our knowledge graph is the first resource provided in this work and contains all of the information in nuScenes as one large graph. It enables further research in trajectory prediction methods with information that was not readily available previously and is a step towards symbolic methods to be explored.

However, exploring neural network models on top of our extensive representation requires a dataset of training pairs. nuScenes and other trajectory prediction datasets are not in this standard form and typically require extensive data preparation. We therefore constructed nSTP, a heterogeneous graph regression dataset for trajectory prediction. It comes in the format of PyTorch Geometric (PyG) [25], which is one of the most widely used graph network libraries. The dataset is readily loadable by PyG dataloaders with input-output pairs of heterogeneous scene graphs and target trajectories. nSTP consists of over 40,000 training pairs.

Formally, a heterogeneous graph $G = (V, E, \tau, \phi)$ has nodes $v \in V$, with node types $\tau(v)$, and edges $(u, v) \in E$, with edge types $\phi(u, v)$. The edges are directed since they are based on properties of the knowledge graph. Each example $i$ in the constructed dataset is a pair $(x_i, y_i) \in (\mathcal{G}, \mathbb{R}^{12})$, where $x_i$ is a scene graph with trajectory information from the past two seconds, local map and target identifier and $y_i$ is the ground truth future trajectory of the target. This makes our dataset a graph regression task. The constraints of 2 seconds into the past and 6 seconds into the future (sampled at 2Hz) are kept from nuScenes, such that any results on our new graph dataset can be compared to those on nuScenes raw data. The training, validation and testing splits from nuScenes are also preserved.

## 4.1. Data-induced inductive bias

The coordinate system used was an important consideration as the right choice of coordinate system enables a data-centric inductive bias to be enforced, namely shift- and rotation-invariance. Inductive biases are widely considered to be essential for deep learning to generalise well [28, 70, 11].

Coordinates in the knowledge graph (and in nuScenes) are initially in a global coordinate system. These were transformed separately for each scene graph into local, scene graph-specific coordinates, with the origin at the location of the target agent and the positive x-axis pointing along the facing direction of the target.

Precisely, let $p_{\text{target}}$ and $R_{\text{target}}$ be the global position (vector) and orientation (rotation matrix) of the target vehicle in scene graph $g$, respectively. Let $p_{\text{global}}$, $R_{\text{global}}$ be arbitrary global position and global orientation, respectively. Their representation in the local frame is given by

$$p_{\text{local}} = R_{\text{target}}^{-1}(p_{\text{global}} - p_{\text{target}}) \tag{21}$$

$$R_{\text{local}} = R_{\text{target}}^{-1} R_{\text{global}} \tag{22}$$

where $R_{\text{target}}^{-1}$ is the inverse of the rotation matrix $R_{\text{target}}$.

This way the coordinates of all entities in $g$ can be transformed into the local coordinate system. Predictions automatically become shift- and rotation-invariant because any shifts and rotations are removed in the transformation. All examples $i$ have the target at the origin, oriented along the positive x-axis. [8] has empirically shown that this transformation improves trajectory prediction performance.

## 4.2. Participant extraction

The trajectory information in a scene graph contains the *Sequence*, *Scene*, *SceneParticipant* and *Participant* nodes as well as the semantic relations between *SceneParticipant*s. The object properties between them in the knowledge graph become heterogeneous edges. The data properties of them are turned into node features. SPARQL queries were used to retrieve the past two seconds of *Scene* instances and the relevant agents in them. Relevant agents are those that may influence the target vehicle's motion, defined as those that are on a piece of relevant extracted map described next. This excludes, for example, scene participants on an opposing

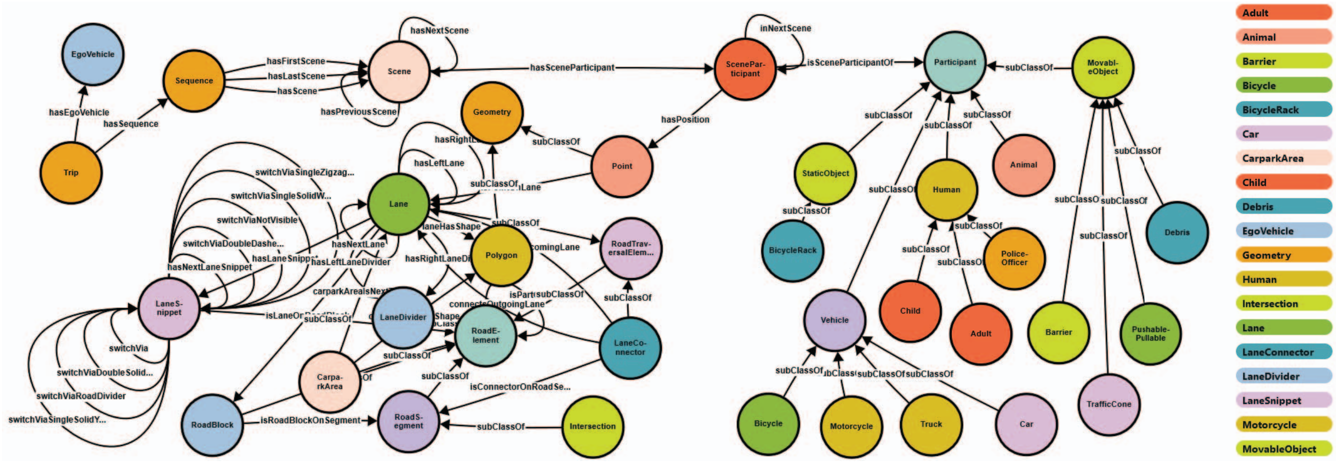


Figure 5. An excerpt of our ontology.

lane from consideration that have already passed the target vehicle, because they are unlikely to influence the target vehicle's future motion.

### 4.3. Map extraction

Besides trajectory information, a scene graph also contains the wealth of map information modelled in our ontology. However, including whole city maps is counterproductive and would make graphs unnecessarily large. The larger a graph, the more long-range dependencies can arise, posing problems for state-of-the-art graph neural networks [23].

Only those parts of the map were considered that affect potential paths of the target. To extract these from the knowledge graph, a target agent is mapped to the road block it is last on, and the *hasNextRoadBlock* edges are followed four times. This is the maximum range travelled within 6 seconds by agents in the nuScenes training data in most cases, as our analyses showed, making this an appropriate heuristic. Adding more into the future than necessary would make the graphs larger than necessary, hurting the performance of current graph neural networks [44]. The map entities surrounding the potential paths are extracted via the explicit spatial relations described in the previous section. These explicit spatial relations are also kept in the heterogeneous scene graphs and, just like all the other object properties are converted into heterogeneous edges.

## 5. Limitations

Our ontology was tailored for representing traffic scenes in the nuScenes dataset. Despite it being a generic traffic model, it is easier to generate an associated knowledge graph from nuScenes than for other raw data sources like Argoverse.

Finally, a limitation of nSTP is that each example's $x_i$ contains between 1,000 and 2,000 nodes on average. GNNs deployed on them need to be able to handle larger graphs, which can be challenging [23].

## 6. Conclusions

A comprehensive ontology for trajectory prediction has been developed with the aim to represent all relevant entities and their spatial and semantic relations in traffic scenes. The ontology has been tailored to the information available in the nuScenes dataset. A knowledge graph based on the ontology has been generated from the nuScenes dataset. The modelled concepts include many elements that were not considered previously, even by state-of-the-art approaches in trajectory prediction. A heterogeneous scene graph dataset was extracted from the knowledge graph, forming the first rich trajectory prediction dataset that can be immediately trained on with neural networks. This included careful pre-processing steps to enforce rotation- and translation-invariance and to only consider agents and map elements that are relevant in each example.

The knowledge graph can be used to investigate how symbolic AI may be incorporated into trajectory prediction models. Reasoning with abstract entities may be a lever to increase robustness and reliability which current deep learning models lack. It is vital to tackle these safety issues to enable deployment of trajectory prediction algorithms in real autonomous vehicles. In addition, the trajectory prediction graph dataset is a major aid to future neural network research for trajectory prediction. It can be used to investigate novel graph neural networks that have access to richer scene information than previous approaches in trajectory prediction.

## 7. Acknowledgements

Many thanks to Benjamin Ruppik for his helpful comments and the Bosch HPC team for providing compute.

# References


[1] Ill-formed maps "singapore-queenstown" and "singapore-hollandvillage" · Issue 862 · nutonomy/nuscenes-devkit — github.com. [Accessed 17-Jul-2023]. 6

[2] Asam openscenario v2.0. Technical report, Assocation for Standardization of Automation and Measuring Systems, 2022. 3

[3] Asam openxontology, concept. Technical report, Assocation for Standardization of Automation and Measuring Systems, 2022. 3

[4] Kartik Ahuja, Yixin Wang, Divyat Mahajan, and Yoshua Bengio. Interventional causal representation learning. *ArXiv*, abs/2209.11924, 2022. 2

[5] S. Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. Dbpedia: A nucleus for a web of open data. In *ISWC/ASWC*, 2007. 3

[6] Mohammadhossein Bahari, Saeed Saadatnejad, Ahmad Nafais Rahimi, Mohammad Shaverdikondori, Mohammad Shahidzadeh, Seyed-Mohsen Moosavi-Dezfooli, and Alexandre Alahi. Vehicle trajectory prediction works, but not everywhere. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17102–17112, 2021. 1

[7] Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinícius Flores Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Çaglar Gülçehre, H. Francis Song, Andrew J. Ballard, Justin Gilmer, George E. Dahl, Ashish Vaswani, Kelsey R. Allen, Charlie Nash, Victoria Langston, Chris Dyer, Nicolas Manfred Otto Heess, Daan Wierstra, Pushmeet Kohli, Matthew M. Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. Relational inductive biases, deep learning, and graph networks. *ArXiv*, abs/1806.01261, 2018. 1

[8] Yannik Benz. Graph-based representations of driving scenarios for vehicle trajectory prediction. Master's thesis, Technische Universitaet Darmstadt, 2022. 7

[9] Hendrik Berkemeyer, Riccardo Franceschini, Tuan Tran, Lin Che, and Gordon Pipa. Feasible and adaptive multimodal trajectory prediction with semantic maneuver fusion. *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8530–8536, 2021. 2

[10] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym, 2016. 1

[11] Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Velivckovic. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *ArXiv*, abs/2104.13478, 2021. 7

[12] Simon Burton, Lydia Gauerhof, and Christian Heinzemann. Making the case for safety of machine learning in highly automated driving. In *SAFECOMP Workshops*, 2017. 2

[13] Martin Volker Butz, Asya Achimova, David K. Bilkey, and Alistair Knott. Event-predictive cognition: A root for conceptual human thought. *Top. Cogn. Sci.*, 13:10–24, 2021. 2

[14] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020. 1

[15] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Sławomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, and James Hays. Argoverse: 3d tracking and forecasting with rich maps. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8740–8749, 2019. 1

[16] Andy Clark. Whatever next? predictive brains, situated agents, and the future of cognitive science. *The Behavioral and brain sciences*, 36 3:181–204, 2013. 2

[17] Henggang Cui, Vladan Radosavljevic, Fang-Chieh Chou, Tsung-Han Lin, Thi Nguyen, Tzu-Kuo Huang, Jeff G. Schneider, and Nemanja Djuric. Multimodal trajectory predictions for autonomous driving using deep convolutional networks. *2019 International Conference on Robotics and Automation (ICRA)*, pages 2090–2096, 2018. 2, 3

[18] Artur S. d'Avila Garcez and L. Lamb. Neurosymbolic ai: The 3rd wave. *ArXiv*, abs/2012.05876, 2020. 2

[19] Hanneke E. M. den Ouden, Peter Kok, and Floris P. de Lange. How prediction errors shape perception, attention, and motivation. *Frontiers in Psychology*, 3, 2012. 2

[20] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 1

[21] Nachiket Deo, Eric M. Wolff, and Oscar Beijbom. Multimodal trajectory prediction conditioned on lane-graph traversals. In *Conference on Robot Learning*, 2021. 3, 5

[22] Nemanja Djuric, Vladan Radosavljevic, Henggang Cui, Thi Nguyen, Fang-Chieh Chou, Tsung-Han Lin, Nitin Singh, and Jeff G. Schneider. Uncertainty-aware short-term motion prediction of traffic actors for autonomous driving. *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 2084–2093, 2018. 2

[23] Vijay Prakash Dwivedi, Ladislav Rampasek, Mikhail Galkin, Alipanah Parviz, Guy Wolf, Anh Tuan Luu, and D. Beaini. Long range graph benchmark. *ArXiv*, abs/2206.08164, 2022. 8

[24] Michael Feld and Christian A. Müller. The automotive ontology: managing knowledge inside the vehicle and sharing it between cars. In *International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, 2011. 3

[25] Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019. 7

[26] Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. Vectornet: Encoding hd maps and agent dynamics from vectorized representation. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11522–11530, 2020. 2, 3

[27] Sebastian Geyer, Marcel Caspar Attila Baltzer, Benjamin Franz, Stephan Hakuli, Michaela Kauer, Martin Kienle,

Sonja Meier, Thomas Weißgerber, Klaus Bengler, Ralph Bruder, Frank Flemisch, and Hermann Winner. Concept and development of a unified ontology for generating test and use-case catalogues for assisted and automated vehicle guidance. *Iet Intelligent Transport Systems*, 8:183–189, 2014. 3

[28] Anirudh Goyal and Yoshua Bengio. Inductive biases for deep learning of higher-level cognition. *CoRR*, abs/2011.15091, 2020. 7

[29] Daniel Grimm, Philip Schörner, Moritz Dressler, and Johann Marius Zöllner. Holistic graph-based motion prediction. *ArXiv*, abs/2301.13545, 2023. 2, 3

[30] Ramanathan V. Guha, Dan Brickley, and Steve Macbeth. Schema.org: Evolution of structured data on the web. *Queue*, 13:10 – 37, 2015. 3

[31] Lavdim Halilaj, Juergen Luettin, Cory Andrew Henson, and Sebastian Monka. Knowledge graphs for automated driving. *2022 IEEE Fifth International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*, pages 98–105, 2022. 3, 4

[32] Manolo Dulva Hina, Clement Thierry, Assia Soukane, and Amar Ramdane-Cherif. Ontological and machine learning approaches for managing driving context in intelligent transportation. In *International Conference on Knowledge Engineering and Ontology Development*, 2017. 3

[33] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, José Emilio Labra Gayo, S. Kirrane, Sebastian Neumaier, Axel Polleres, Roberto Navigli, Axel-Cyrille Ngonga Ngomo, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan Sequeda, Steffen Staab, and Antoine Zimmermann. Knowledge graphs. *Communications of the ACM*, 64:96 – 104, 2020. 1, 2

[34] Yingjie Hu, Krzysztof Janowicz, David Carral, Simon Scheider, Werner Kuhn, Gary Berg-Cross, Pascal Hitzler, Mike Dean, and Dave Kolas. A geo-ontology design pattern for semantic trajectories. In *Conference On Spatial Information Theory*, 2013. 3

[35] Road vehicles – Functional safety. Standard, International Organization for Standardization, Geneva, CH, Mar. 2018. 2

[36] Krzysztof Janowicz, Armin Haller, Simon J. D. Cox, Danh Le-Phuoc, and Maxime Lefrançois. Sosa: A lightweight ontology for sensors, observations, samples, and actuators. *J. Web Semant.*, 56:1–10, 2018. 3

[37] Shaoxiong Ji, Shirui Pan, E. Cambria, Pekka Marttinen, and Philip S. Yu. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 33:494–514, 2020. 1

[38] Xiaosong Jia, Peng Wu, Li Chen, Hongyang Li, Yu Sen Liu, and Junchi Yan. Hdgt: Heterogeneous driving graph transformer for multi-agent trajectory prediction via scene encoding. *ArXiv*, abs/2205.09753, 2022. 3

[39] Benjamin Klotz, Raphael Troncy, Daniel Wilms, and Christian Bonnet. Vsso: The vehicle signal and attribute ontology. In *SSN@ISWC*, 2018. 3

[40] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995. 2

[41] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86:2278–2324, 1998. 1

[42] Jiachen Li, Fan Yang, Masayoshi Tomizuka, and Chiho Choi. Evolvegraph: Multi-agent trajectory prediction with dynamic relational reasoning. *arXiv: Computer Vision and Pattern Recognition*, 2020. 2

[43] Longyuan Li, Jinhui Yao, Li Kevin Wenliang, Tongze He, Tianjun Xiao, Junchi Yan, David Paul Wipf, and Zheng Zhang. Grin: Generative relation and intention network for multi-agent trajectory prediction. In *Neural Information Processing Systems*, 2021. 2

[44] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *AAAI Conference on Artificial Intelligence*, 2018. 8

[45] Xin Li, Xiaowen Ying, and Mooi Choo Chuah. Grip: Graph-based interaction-aware trajectory prediction. *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 3960–3966, 2019. 2

[46] Ming Liang, Binh Yang, Rui Hu, Yun Chen, Renjie Liao, Song Feng, and Raquel Urtasun. Learning lane graph representations for motion forecasting. *ArXiv*, abs/2007.13732, 2020. 2, 3

[47] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, 2014. 1

[48] Zachary Chase Lipton. The mythos of model interpretability. *Queue*, 16:31 – 57, 2016. 1

[49] Mengmeng Liu, Hao Cheng, Lin Chen, Hellward Broszio, Jiangtao Li, Runjiang Zhao, Monika Sester, and Michael Ying Yang. Laformer: Trajectory prediction for autonomous driving with lane-aware scene constraints. *arXiv preprint arXiv:2302.13933*, 2023. 2, 3

[50] Juergen Luettin, Sebastian Monka, Cory Andrew Henson, and Lavdim Halilaj. A survey on knowledge graph-based methods for automated driving. In *Iberoamerican Conference on Knowledge Graphs and Semantic Web*, 2022. 3

[51] Yuexin Ma, Xinge Zhu, Sibo Zhang, Ruigang Yang, Wenping Wang, and Dinesh Manocha. Trafficpredict: Trajectory prediction for heterogeneous traffic-agents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6120–6127, 2019. 1

[52] Gary F. Marcus. Deep learning: A critical appraisal. *ArXiv*, abs/1801.00631, 2018. 1

[53] Sebastian Monka, Lavdim Halilaj, and Achim Rettinger. A survey on visual transfer learning using knowledge graphs. *ArXiv*, abs/2201.11794, 2022. 1

[54] Patel-Schneider P.F. Parsia B. Motik, B. Owl 2 web ontology language: Document overview (second edition). `https://www.w3.org/TR/owl2-overview/`, 2012. 4

[55] Tung Phan-Minh, Elena Corina Grigore, Freddy A. Boulton, Oscar Beijbom, and Eric M. Wolff. Covernet: Multimodal behavior prediction using trajectory sets. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14062–14071, 2019. 2, 3

[56] Fabian Poggenhans, Jan-Hendrik Pauls, Johannes Janosovits, Stefan Orf, Maximilian Naumann, Florian Kuhnt, and Matthias Mayr. Lanelet2: A high-definition map framework for the future of automated driving. In *2018 21st international conference on intelligent transportation systems (ITSC)*, pages 1672–1679. IEEE, 2018. 5

[57] Norma C. Presmeg. The body in the mind: The bodily basis of meaning, imagination and reason. *Educational Studies in Mathematics*, 23:307–314, 1992. 2

[58] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *ArXiv*, abs/1606.05250, 2016. 1

[59] Sohail Sarwar, Saad Zia, Zia Ul-Qayyum, Muddesar Iqbal, Muhammad Safyan, Shahid Mumtaz, Raúl García-Castro, and Konstantin Kostromitin. Context aware ontology-based hybrid intelligent framework for vehicle driver categorization. *Transactions on Emerging Telecommunications Technologies*, 33, 2019. 3

[60] Nino Scherrer, Anirudh Goyal, Stefan Bauer, Yoshua Bengio, and Nan Rosemary Ke. On the generalization and adaption performance of causal models. *ArXiv*, abs/2206.04620, 2022. 2

[61] Bernhard Scholkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Toward causal representation learning. *Proceedings of the IEEE*, 109:612–634, 2021. 2

[62] Christopher Summerfield and Tobias Egner. Expectation (and attention) in visual cognition. *Trends in Cognitive Sciences*, 13:403–409, 2009. 2

[63] Yogita Suryawanshi, Haonan Qiu, Adel Ayara, and Birte Glimm. An ontological model for map data in automotive systems. *2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*, pages 140–147, 2019. 3

[64] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, D. Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2013. 1

[65] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012. 1

[66] Simon Ulbrich, Till Menzel, Andreas Reschka, Fabian Schuldt, and Markus Maurer. Defining and substantiating the terms scene, situation, and scenario for automated driving. *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pages 982–988, 2015. 3

[67] Simon Ulbrich, Tobias Nothdurft, Markus Maurer, and Peter Hecker. Graph-based context representation, environment modeling and information aggregation for automated driving. *2014 IEEE Intelligent Vehicles Symposium Proceedings*, pages 541–547, 2014. 3

[68] Balakrishnan Varadarajan, Ahmed S. Hefny, Avikalp Srivastava, Khaled S. Refaat, Nigamaa Nayakanti, Andre Cornman, K. M. Chen, Bertrand Douillard, C. P. Lam, Drago Anguelov, and Benjamin Sapp. Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction. *2022 International Conference on Robotics and Automation (ICRA)*, pages 7814–7821, 2021. 2

[69] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *ArXiv*, abs/1804.07461, 2018. 1

[70] Max Welling. Do we still need models or just more data and compute? *University of Amsterdam*, 2019. 7

[71] Lukas Westhofen, Christian Neurohr, Martin Butz, Maike Scholtes, and Michael Schuldes. Using ontologies for the formalization and recognition of criticality for automated driving. *arXiv preprint arXiv:2205.01532*, 2022. 3

[72] Wei Zhan, Liting Sun, Di Wang, Haojie Shi, Aubrey Clausse, Maximilian Naumann, Julius Kümmerle, Hendrik Königshof, Christoph Stiller, Arnaud de La Fortelle, and Masayoshi Tomizuka. Interaction dataset: An international, adversarial and cooperative motion dataset in interactive driving scenarios with semantic maps. *ArXiv*, abs/1910.03088, 2019. 1

[73] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *ArXiv*, abs/1611.03530, 2016. 1

[74] Lihua Zhao, Ryutaro Ichise, Seiichi Mita, and Yutaka Sasaki. Core ontologies for safe autonomous driving. In *International Workshop on the Semantic Web*, 2015. 3

[75] Maximilian Zipfl, Felix Hertlein, Achim Rettinger, Steffen Thoma, Lavdim Halilaj, Juergen Luettin, Stefan Schmid, and Cory Andrew Henson. Relation-based motion prediction using traffic scene graphs. *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, pages 825–831, 2022. 2, 3, 4