# GoRela: Go Relative for Viewpoint-Invariant Motion Forecasting

**Alexander Cui, Sergio Casas, Kelvin Wong, Simon Suo, Raquel Urtasun**
Waabi, University of Toronto
{acui, sergio, kwong, ssuo, urtasun}@waabi.ai

***Abstract*— The task of motion forecasting is critical for self-driving vehicles (SDVs) to be able to plan a safe maneuver. Towards this goal, modern approaches reason about the map, the agents' past trajectories and their interactions in order to produce accurate forecasts. The predominant approach has been to encode the map and other agents in the reference frame of each target agent. However, this approach is computationally expensive for multi-agent prediction as inference needs to be run for each agent. To tackle the scaling challenge, the solution thus far has been to encode all agents and the map in a shared coordinate frame (e.g., the SDV frame). However, this is sample inefficient and vulnerable to domain shift (e.g., when the SDV visits uncommon states). In contrast, in this paper, we propose an efficient shared encoding for all agents and the map without sacrificing accuracy or generalization. Towards this goal, we leverage pair-wise relative positional encodings to represent geometric relationships between the agents and the map elements in a heterogeneous spatial graph. This parameterization allows us to be invariant to scene viewpoint, and save online computation by re-using map embeddings computed offline. Our decoder is also viewpoint agnostic, predicting agent goals on the lane graph to enable diverse and context-aware multimodal prediction. We demonstrate the effectiveness of our approach on the urban Argoverse 2 benchmark as well as a novel highway dataset. For more information, visit the project website: https://waabi.ai/research/gorela**

## I. Introduction

Predicting the future motion of the traffic participants is critical for SDVs. To drive safely, predictions have to be not only accurate and generalize across many scenarios, but also made in a timely manner so that the SDV can react appropriately.

Existing methods have made compromises in their accuracy and generalization abilities vs. computation needs. Most approaches [1]–[3] have prioritized accuracy and generalization at the expense of runtime by processing the scene from the viewpoint of each target agent. Unfortunately this approach does not scale to situations with a large number of agents, which arise in crowded urban scenes or in highways where, due to the agents' high speed, predictions need to be performed over a very large area, thus potentially containing many agents. Other works [4]–[6] have focused on achieving a reasonable inference time at the expense of less accurate models that are less capable of generalization. These works propose a shared encoding of the scene for all agents by using a fixed viewpoint for all predictions such as the coordinate frame defined by the SDV's current pose. While achieving low latency, the predictions are no longer invariant to the viewpoint from which the scene is encoded, and as a consequence may not generalize to rarely seen or novel SDV poses (e.g., when the SDV is performing a U-turn).

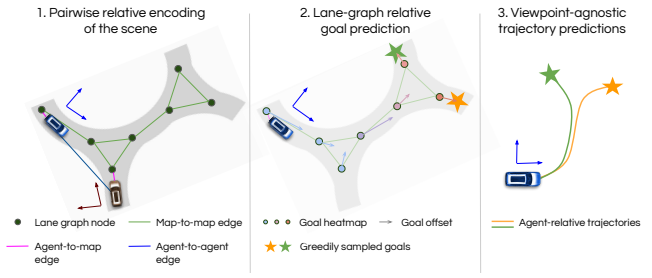


Fig. 1. Our model, GoRela, encodes the scene by reasoning about pair-wise relative geometric relationships between the agents and the lane graph with a GNN. Then, it predicts a goal distribution using the lane graph nodes as anchors, and samples goals from it. Finally, trajectories are generated conditioned on the goals. The architecture is viewpoint-invariant, thus allowing for shared computation across agents.

In this paper, we propose to encode the interaction between different map entities and the agents in a viewpoint agnostic representation by modelling their pairwise relative relationships. This viewpoint invariance provides several key advantages: (i) it helps the model generalize by greatly reducing the space of the problem domain, (ii) it makes learning more sample efficient, making training faster while requiring less data (e.g., removing the need for any data augmentation during training in the form of viewpoint translation and rotation), (iii) it keeps inference highly resource-efficient as the scene encoding (which is the heaviest module) only needs to be processed once, and (iv) it enables caching of any computation that depends solely on the static parts of the scene, allowing our model to compute the map embeddings offline, thus saving critical processing time on the road. Through extensive quantitative evaluation on both urban and highway datasets, we demonstrate that our model is more effective, generalizes better to novel viewpoints, and it is less data hungry than competing methods.

We structure the paper as follows: Section II offers a survey of motion forecasting methods, Section III introduces a new pair-wise relative heterogeneous graph neural network that is the backbone for interaction reasoning in our motion forecasting model, presented in Section IV. Finally, we benchmark our approach in Section V.

## II. Related Work in Motion Forecasting

In this section we review prior approaches, breaking down their contributions to several motion forecasting modules.

**Agent history encoding:** Most works represent the past trajectory as a sequence of 2D waypoints in a global frame.

Rasterizing the past trajectory and feeding it through a 2D convolutional neural network (CNN) was first proposed [3], [7], [8]. Then, fully vectorized methods [2], [5], [9] proposed to transform all past poses to a common coordinate frame and process them with 1D CNNs, recurrent neural networks (RNN) or Transformers.

**Map encoding:** Early works encode the map using bird's-eye view (BEV) rasters and 2D CNNs [3], [8], [10]–[13]. However, this design is poorly suited when the SDV heading is not aligned with the road as the encoding is not rotationally invariant. In self-driving datasets, the SDV heading will generally align with its lane, and thus heading offsets will make the inputs out of distribution regardless of the behavior of other agents. Processing a region of interest (RoI) centered and rotated around each agent [7], [14]–[19] mitigates this issue, but makes inference computationally demanding as these RoIs need to be large. Recently, vector-based approaches represent the map as a set of lanes and nodes, employing graph neural networks (GNNs) or Transformers to fuse map features with the agents. To encode the scene context, [2], [9], [20]–[22] transform the map elements' coordinates to the target agent frame. To enable multi-agent prediction while being invariant to SDV pose, many follow [1] and repeat the scene encoding for each agent, linearly increasing the cost as the number of agents grow. To tackle the linear growth in computational cost, [5], [6], [15], [16], [23], [24] encode the scene in a global frame shared by all agents, but lose viewpoint invariance, thus sacrificing accuracy and generalization and making the model more data hungry. In contrast, our lane-graph encoder achieves a shared, viewpoint-invariant scene encoding.

**Heterogeneous fusion:** To fuse information between agents and map, LaneGCN[9] defines a late fusion where information is propagated in the following order: agent-to-map, map-to-map, map-to-agent, agent-to-agent. LaneRCNN[6] instead early fuses agents' past trajectories into the lane-graph nodes. In an attempt to simplify the pipeline, SceneTransformer [4] regards agent waypoints and lane-graph nodes as tokens and performs multiple rounds of cross-attention. Similarly to the map encoding, a shared coordinate frame is assumed in order to process the scene only once, at the expense of losing invariance to the viewpoint. Instead, our model leverages edge attributes in a heterogeneous graph, where geometric information is expressed as pair-wise relative positional encodings, thus being viewpoint invariant while maintaining the desired shared scene processing across agents. As far as we know, concurrent work HDGT[25] is the only other method to also leverage a pair-wise relative encoding. However, our work presents several advantages: (i) our bounded pair-wise relative positional encoding removes the need for many graph creation heuristics, (ii) our goal-based trajectory decoder is more powerful than direct regression (as shown by our ablations), and (iii) it enables map embedding caching due to its late fusion of agents and map, thus reducing model latency. These differences contribute to a superior performance in Argoverse 2 w.r.t. HDGT [25].

**Future trajectory decoding:** Regressing all the parameters of a mixture of future trajectories in parallel [7], [9], [22], [23], [25] has been the default approach to trajectory decoding due to its simplicity. However, the trajectories output by these methods tend to go off-map unless prior knowledge losses are imposed as soft constraints [26]. While methods that model the future trajectory autoregressively over time [27], [28] can help with better map understanding, they bring the additional issue of compounding errors [29]. Many state-of-the-art methods today factorize trajectory prediction into goal (endpoint) prediction and trajectory completion [5], [6], [30]–[34]. The intuition is that the goal captures most of the stochasticity in the future and also allows for simple multimodal reasoning. Since runtime is not evaluated in motion forecasting benchmarks, ensembles of multiple models followed by clustering [20], [35], [36] have emerged in order to predict a better covering set of modes. However, this is prohibitive for autonomy where reaction time is critical. In our method, we use goal-based prediction anchored to lane graph nodes, and propose a simple yet effective greedy goal sampler that is competitive without incurring any expensive postprocessing such as ensemble clustering.

## III. Pair-wise Relative Heterogeneous GNN

GNNs process graph structures by exchanging messages between node pairs connected by edges. In this section, we propose a novel GNN architecture that will be extensively used in the next section as the backbone for encoding graphs composed of both agents and map nodes, and decoding future predictions from them. The goal of our architecture is to capture complex interactions between nodes in a heterogeneous spatial graph. By spatial, we mean that every node $\boldsymbol{x}_i$ has both a feature vector with general attributes $\boldsymbol{x}_i^a$ as well as a 3-DOF pose $\boldsymbol{x}_i^p$ describing its centroid and yaw. Heterogeneous here means that nodes may belong to different semantic categories, and their node attributes may have different dimensionality. This graph neural network models directional relationships via continuous edge attributes $\boldsymbol{e}_{i\to j}^a$ and discrete edge classes $\boldsymbol{e}_{i\to j}^c$. In the next section, we describe the parameterization of the positional encodings we employ to describe the relative pose between two nodes, which make learning easier by being magnitude bounded and viewpoint-invariant (i.e., agnostic to reference frame rotations and translations). We then explain our heterogeneous message passing layer, which exploits these encodings as edge attributes. We refer the reader to Fig. 2 for an illustration of this layer and the pair-wise relative positional encodings.

**Pair-wise relative positional encoding (PairPose):** Each node in the graph has a pose $\boldsymbol{x}_i^p$, which is composed of a centroid $\boldsymbol{c_i}$ and a unit vector in the heading direction $\boldsymbol{h_i}$. To represent the directional, pairwise relationship between node $i$ and $j$ (i.e., $i \to j$), we first compute the displacement vector between each node's centroids $\boldsymbol{v}_{i\to j} = \boldsymbol{c}_i - \boldsymbol{c}_j$ as well as the sine and cosine of the heading difference

$$\sin(\alpha_{i\to j}) = \boldsymbol{h}_i \times \boldsymbol{h}_j, \qquad \cos(\alpha_{i\to j}) = \boldsymbol{h}_i \cdot \boldsymbol{h}_j$$

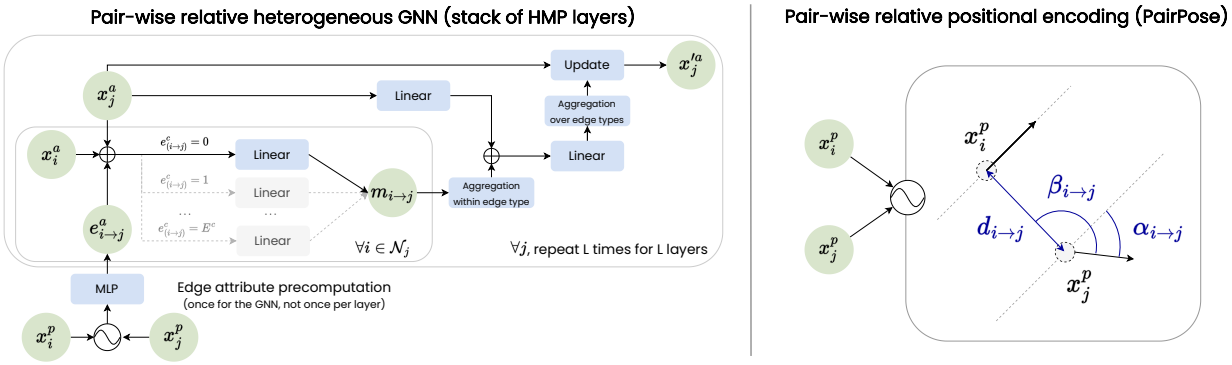


Fig. 2. Left: **Heterogeneous message passing** between two nodes. Right: Zoom-in to visually explain our **pair-wise relative positional encoding**.

Note that the displacement vector $\boldsymbol{v}_{i\to j}$ depends on the arbitrary global frame the centroids are expressed in, and thus is not viewpoint invariant. To achieve invariance, we instead utilize the centroid distance $d_{i\to j} = ||\boldsymbol{v}_{i\to j}||_2$, together with the sine and cosine of the angle between the displacement vector $\boldsymbol{v}_{i\to j}$ and the heading $\boldsymbol{h}_j$ (shown in Fig. 2-Right)

$$\sin(\beta_{i\to j}) = \frac{\boldsymbol{v}_{i\to j} \times \boldsymbol{h}_j}{|\boldsymbol{v}_{i\to j}||\boldsymbol{h}_j|}, \cos(\beta_{i\to j}) = \frac{\boldsymbol{v}_{i\to j} \cdot \boldsymbol{h}_j}{|\boldsymbol{v}_{i\to j}||\boldsymbol{h}_j|}$$

To make the centroid distances bounded, we follow [37] and map each distance to a vector $\boldsymbol{p}_{i\to j} = [p_1, \ldots, p_N, r_1, \ldots, r_N]$ composed of sine and cosine functions of $N$ different frequencies that represent the range of distances that we are interested in (from a handful of meters to hundreds of meters). More concretely,

$$p_n = \sin(d_{i\to j} \exp(\frac{4n}{N})), \qquad r_n = \cos(d_{i\to j} \exp(\frac{4n}{N}))$$

The pair-wise geometric relationship of entities $i$ and $j$ can be summarized as a concatenation ($\oplus$):

$$\boldsymbol{g}^a_{i\to j} = [\sin(\alpha_{i\to j}), \cos(\alpha_{i\to j}), \sin(\beta_{i\to j}), \cos(\beta_{i\to j})] \oplus \boldsymbol{p}_{i\to j}$$

The final positional encoding is then learned as

$$\boldsymbol{e}^a_{i\to j} = \text{MLP}(g^a_{i\to j})$$

**Heterogeneous message passing (HMP) layer:** The goal of this layer is to update the node features $\boldsymbol{x}^a_j$ in a directed graph by taking into account all its incoming neighbors' features $\boldsymbol{x}^a_i \in \mathcal{N}_j$ as well as their pair-wise relative positional encodings $\boldsymbol{e}_{i\to j}$. The message passing process is depicted in Fig. 2-Left. For every neighbor $i \in \mathcal{N}_j$, we compute a message $m_{i\to j}$ by linearly projecting the concatenation of the source features $\boldsymbol{x}^a_i$ and edge attributes $\boldsymbol{e}^a_{i\to j}$. Note that the weights of this linear layer are different depending on the discrete edge class $\boldsymbol{e}^c_{i\to j}$, and that the dimensionality of those weights vary by node class. Then, all incoming messages for the same edge type are aggregated using a permutation invariant aggregation function (e.g., max-pooling). Next, aggregated messages are concatenated with a linear projection of the node features, and fused with another linear layer. Then, the message for each edge type is aggregated. Finally, an update function (e.g., GRU cell) enables our heterogeneous message passing layer to keep or forget information from the previous features $\boldsymbol{x}^a_j$ for the updated features $\boldsymbol{x}'^a_j$. We compose our pair-wise relative heterogeneous GNN by stacking multiple HMP layers. We do not use GAT [38] as it only uses the edge to compute attention weights over neighboring nodes, and does not use the edge embedding to directly update the node embedding, which limits its model capacity.

## IV. Viewpoint-Invariant Motion Forecasting

In this section, we describe how we efficiently and effectively forecast multi-agent future trajectories based on the map as well as past trajectories. Towards this goal, we first compute agent features and map features separately because (i) the nature of the features is different (spatio-temporal vs. spatial), and (ii) efficiency - the map features can be computed offline as they only rely on the high-definition maps. We then leverage the heterogeneous message passing (HMP) layers proposed in Section III to model inter- and intra-class interactions on a single heterogeneous graph that connects agents and map elements. Finally, we predict the future trajectory of each agent by prediction first their goal (waypoint at the end of the prediction horizon), and then their full trajectory given the goal. Fig. 3 shows an illustration of our approach. We dub our method GoRela to highlight the fact that we "go relative" in the modelling of all pair-wise geometric interactions, achieving a viewpoint agnostic architecture. In the remainder of this section, we describe every module and the training objective.

**Agent history encoder:** The spatial input features of current multi-agent prediction systems are typically encoded in a coordinate frame whose origin is the SDV position and the x-axis is aligned with its heading [4], [5], [9]. This has the undesired property that the same trajectory has different embeddings if the SDV is at different positions and or orientations in the scene. Instead, we design a viewpoint-invariant representation to express the past trajectory as a sequence of pair-wise relative positional encodings between the past waypoints and the current pose. More precisely, we encode the past trajectory as $\tau = [\boldsymbol{e}_{-T\to 0}, \ldots, \boldsymbol{e}_{-1\to 0}] \in \mathbb{R}^{T\times D}$, where $\boldsymbol{e}_{t\to 0}$ refers to the pair-wise relative positional encoding of the pose $x^p_t$ at a past time step $t < 0$ with respect to the current pose $x^p_0$ (as described in Eq. **??**). Note that current time is $t = 0$, negative times denote the past with $t = -T$ being the history horizon, and $D$ is the dimensionality of our pair-wise relative positional

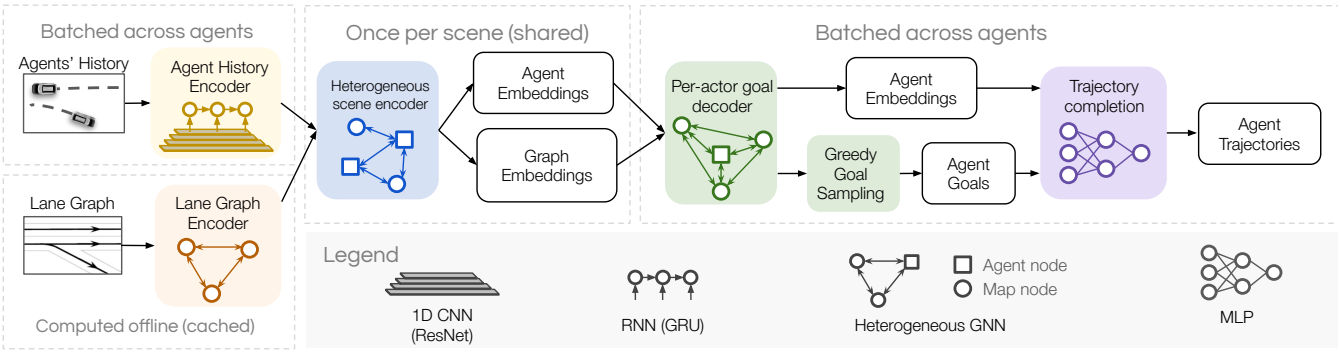


Fig. 3. **GoRela model pipeline**. Thanks to its viewpoint invariance, our model can run the lane-graph encoder offline and cache its activations for online runtime savings, and perform heterogeneous scene encoder just once, shared across all agents.

encoding. We then pass each agent's features into a 1D convolutional network with residual connections followed by a gated recurrent unit (GRU), and use the final hidden state of the GRU as the embedding of the agent's trajectory. Intuitively, this works well because the 1D CNN can extract local temporal patterns and the GRU can aggregate them in a learnable hidden state. Importantly, the computation for all agents can be batched and thus inference is very efficient even in complex crowded scenes.

**Lane-graph encoder:** We build our lane graph by sampling the centerlines in the high-definition map at regular intervals (e.g., $3m$ for urban, $10m$ for highway) to obtain lane segments. Each lane segment is represented as a node in the graph containing features such as its length, width, curvature, speed limit, and lane boundary type (e.g., solid, dashed). Following [9], these lane nodes are then connected with 4 different relationships: successors, predecessors, left and right neighbors. For successors and predecessors, we also include dilations (i.e., skip connections) for $\{2, 3, 4, 5\}$ hops in order to make the receptive field of our GNN grow faster over layers, which helps particularly with fast moving vehicles. On top of these, we find it helpful to sample map nodes uniformly from crosswalk polygons since these are highly relevant for pedestrians and vehicles interacting with pedestrians. To understand the interactions occurring at overlapping map entities (e.g., lanes at intersections, crosswalks and lanes) we add a "conflict" edge for every pair of nodes that are less than 2.5 meters apart and belong to distinct map entities. We use a stack of the HMP layers to update the embeddings of every node in the graph. Importantly, our viewpoint agnostic lane graph encoder allows us to compute map embeddings offline for large-scale maps after training GoRela. Thanks to its memory-efficiency and coordinate frame invariance, we can compute the map embeddings for a large tile offline in a single inference pass over the lane-graph encoder, eliminating the need for stitching regions of map embeddings together. On the onboard side, the cached lane graph embeddings are served to autonomy, improving the SDV reaction time.

**Heterogeneous scene encoder:** We fuse the agent and lane graph features using another stack of HMP layers. The heterogeneous scene graph contains agents and map nodes, whose input attributes are initialized with the results of the previously described agent history encoder and lane graph encoder. In addition to the map-to-map edges used in the lane graph encoder, we include agent-to-agent, agent-to-map and map-to-agent edges in the graph. We create agent-to-map and map-to-agent edges between agents and the lane graph node that is nearest to their position at $t = 0$. For agent-to-agent, we connect every pair of agent nodes that is closer than 100 meters. At every round of message passing, we compute messages for all edge types, and update both the agent and lane graph node embeddings with heterogeneous messages incoming from the different edge types. This is different from LaneGCN[9], which defines a custom ordering for message passing of different types. Unlike HEAT's [22] message passing, our HMP layers use edge functions with specialized parameters for each edge class for improved expressivity.

**Goal-based decoder:** To decode agent future trajectories, we first predict their goal (last waypoint). We cast the goal prediction as a multi-class classification problem over lane graph nodes [6], as this allows us to leverage the rich map embeddings computed by the heterogeneous scene encoder. To achieve a resolution beyond that of the lane-graph, we also regress a continuous offset for each candidate goal with respect to its lane-graph node anchor. To predict these classification scores and regression offsets, we create a graph composed of $A$ connected components, one for each agent in the scene. Each connected component contains one actor and a copy of all the map nodes. Regarding the graph connectivity, we preserve the same map-to-map edges as in upstream components, and connect all possible map-to-actor and actor-to-map pairs. The agent and map node features are initialized to the outputs of the heterogeneous scene encoder, and they are updated by a stack of HMP layers.

**Greedy goal sampler:** Predicting multi-modal trajectories is essential for the motion planner to be safe with respect to any future that might unfold. To predict $K$ modes, we propose a simple yet effective greedy goal sampler. At every iteration, we (1) sample the goal with the highest probability, (2) remove every node closer than $\gamma$ meters, (3) downweight

| Split | Model | BrierFDE K=6 | FDE K=6 | ADE K=6 | MR K=6 | FDE K=1 | ADE K=1 | MR K=1 |
|---|---|---|---|---|---|---|---|---|
| Test (focal agent) | THOMAS [5] | 2.16 | 1.51 | 0.88 | **0.20** | 4.71 | 1.95 | **0.64** |
| | GoRela | **2.01** | **1.48** | **0.76** | 0.22 | **4.62** | **1.82** | 0.66 |
| Validation (multi-agent: focal+scored) | MultiPath*[7] | 2.54 | 2.13 | 0.89 | 0.33 | 14.90 | 7.06 | 0.52 |
| | MTP*[3] | 2.05 | 1.54 | 0.68 | 0.24 | 6.66 | 2.74 | 0.47 |
| | LaneGCN[9] | 1.94 | 1.34 | 0.55 | 0.22 | 4.82 | 1.82 | 0.51 |
| | SceneTransformer*[4] | 1.80 | 1.24 | 0.52 | 0.20 | 4.57 | 1.75 | 0.46 |
| | GoRela | **1.29** | **0.96** | **0.42** | **0.14** | **2.43** | **0.95** | **0.32** |

TABLE I: **Comparison against state of the art on Argoverse 2**. All metrics are minimum error over $K$ modes (lower is better). * indicates re-implementation as these methods are not open-sourced.

| Model | BrierFDE K=6 | BrierATE K=6 | BrierCTE K=6 | ATE K=1 | CTE K=1 |
|---|---|---|---|---|---|
| MultiPath*[7] | 5.37 | 2.42 | 0.68 | 17.07 | 0.86 |
| MTP*[3] | 3.43 | 1.64 | 0.88 | 5.08 | 0.59 |
| LaneGCN[9] | 3.78 | 1.31 | 0.89 | 4.67 | 0.81 |
| SceneTransformer*[4] | 3.17 | 1.51 | 1.15 | 3.56 | 0.71 |
| GoRela | **2.63** | **1.14** | **0.57** | **2.27** | **0.40** |

TABLE II: **Comparison against state of the art on HighwaySim**. All metrics are minimum error over $K$ modes.

| Model† | $\mathcal{E}$ | $\mathcal{D}$ | $\mathcal{G}$ | $e_{i\to j}$ | BrierFDE K=6 | FDE K=6 | FDE K=1 |
|---|---|---|---|---|---|---|---|
| $M_1$ | ✗ | ✓ | ✓ | ✓ | 1.53 | 1.18 | 3.52 |
| $M_2$ | ✓ | ✗ | ✗ | ✓ | 2.19 | 1.59 | 5.67 |
| $M_3$ | ✓ | ✓ | ✗ | ✓ | 1.65 | 1.21 | **2.77** |
| $M_4$ | ✓ | ✓ | ✓ | ✗ | 1.89 | 1.43 | 3.44 |
| GoRela | ✓ | ✓ | ✓ | ✓ | **1.45** | **1.08** | **2.77** |

TABLE III: **Ablation study** on Argoverse 2 (val). We ablate the scene encoder $\mathcal{E}$, goal-based decoder $\mathcal{D}$, greedy sampler $\mathcal{G}$, and PairPose in HMP $e_{i\to j}$. † Models trained on 25% of the training set.

every node closer than $\nu$ ($\nu > \gamma$) meters by a factor of $\zeta$. We repeat this process for $K$ iterations. The intuition is that if the probability distribution is uni-modal we should draw as many samples as possible from it, while if the distribution is multi-modal (e.g., agent at intersection), it is desirable to sample from different modes even if the probability mass around each mode is imbalanced.

**Trajectory completion:** Finally, we perform trajectory completion to all goals in parallel. We perform the trajectory prediction in agent-relative coordinates such that the network can leverage the prior that vehicles initially progress along their heading direction (i.e., the x-axis). We use a shared MLP across agent classes (vehicles, pedestrians and cyclists) that predicts the sequence of 2D waypoints as a flat vector in $\mathbb{R}^{2T_f}$, where $T_f$ is the prediction horizon. We find that sharing the MLP layers across agent classes is beneficial as there are some priors that all trajectories follow, which is beneficial for learning in class-imbalanced settings.

**Training:** We optimize a multi-task loss, which is a linear combination of 3 terms: goal classification, goal regression and trajectory completion. We employ focal loss [39] for goal classification, serving as a form of hard example mining. We supervise the offset regression only for the node that is closest to the goal, and use a Huber loss in node frame. We train our trajectory completion using teacher forcing, meaning that during training we feed the ground-truth goal to the trajectory completion module. This is beneficial since our goal predictions may not always capture the right mode for the goal, particularly at early stages of training. We employ a simple Huber loss on each waypoint coordinate in agent-centric frame. Our final model was trained for 17 epochs, using 16 T4 GPUs for a total batch size of 64. We use Adam optimizer with a learning rate 5.e-4, with a step-wise scheduler with 0.25 decay and 15 step-size.

## V. Experimental Evaluation

In this section, we first show that our method outperforms the state-of-the-art in both urban and highway benchmarks. We then show through a comprehensive ablation study that each component of the model contribute to our improvements. Finally, we show qualitative results. Please see our supplementary materials for implementation details, a quantitative study of the advantages of viewpoint invariance when perturbing the SDV heading, an analysis of how runtime scales with number of actors, and experiments on the sample efficiency when using pair-wise relative positional encoding vs. geometric features in a global coordinate frame.

**Datasets:** We use two complementary datasets to evaluate how well our method works in both urban and highway domains, which present different challenges. Argoverse 2 [40] includes 250,000 city traffic scenarios where trajectories must be predicted for the future 6s, given 5s of history at 10Hz and high-definition maps. The main difficulty is characterizing the multi-modality of the future trajectory distribution due to (i) complex road topologies and (ii) multi-agent interactions, since the scenarios were mined for such cases. For evaluation, agents are divided into focal (one per scenario, hard examples), scored (interactive agents with focal), and unscored (background agents that are not prediction targets). Our second benchmark is HighwaySim, a novel simulated dataset composed of 70,000 frames of object tracks, including vehicles and motorcycles. We generated this dataset with an internal, state-of-the-art simulator using reactive actors and a variety of highway map topologies such as curved roads, forks and merges. Models are tasked with predicting a 6s trajectory at 2Hz, using 0.5s of history at 10Hz for all agents in the scenario. The much higher speeds at which vehicles travel stress the need for a large receptive field, which in turn demands an efficient architecture. Since the SDV is also traveling at high speeds, predicting precise trajectories in terms of lateral deviations from lane centerlines is critical to avoid unsafe harsh brakes.

**Comparison against state-of-the-art:** Table I shows our results in Argoverse 2. Our method outperforms all published

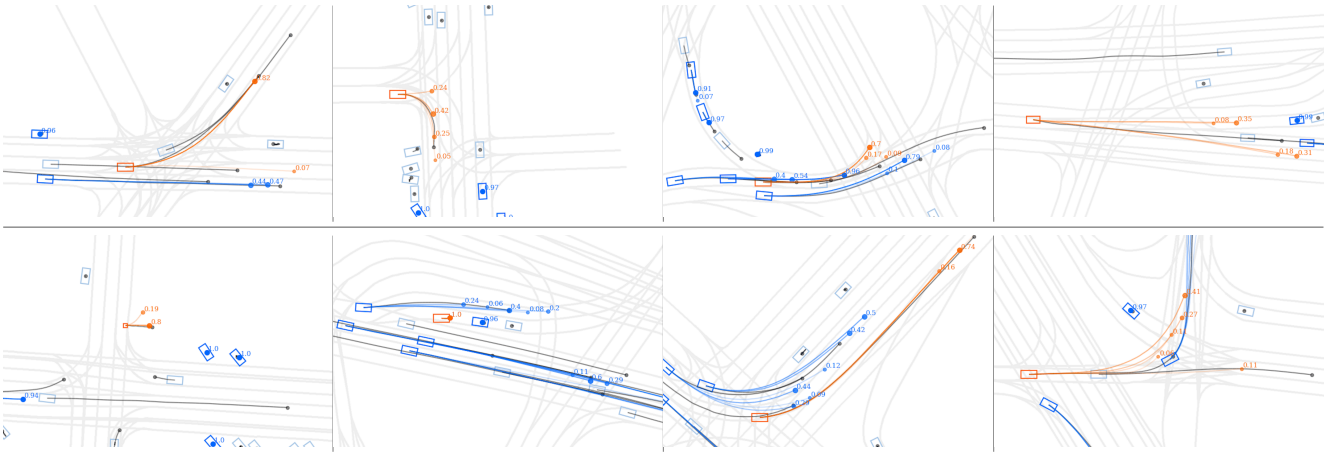

Fig. 4. **Qualitative results in Argoverse 2**. 6 seconds long multi-modal motion forecasts for focal and scored agents. Ground-truth plotted in gray.

methods in the leaderboard (test set), which evaluates the prediction quality on difficult-to-forecast agents (focal) in highly interactive scenarios. Our model performs particularly well on the ranking metric BrierMinFDE, which evaluates not only how close the best mode is to the ground-truth trajectory, but also how much confidence is placed on it. This indicates that our predicted mode probabilities are better calibrated [41] than the baselines'. We refer the reader to the evaluation website[1] for further details about the metrics. We also show large gains in the validation set over various seminal works that are not available on the leaderboard. In this experiment, we evaluate multiple agents per scenario, which we care about for the downstream application of self-driving, where the focal and scored agents are evaluated. We note that the baselines process the scene in the frame of the focal agent as they are not viewpoint invariant.

Table II showcases the results on HighwaySim. Due to the greater difference between longitudinal and lateral motion in the highway, it is important to break down the prediction error into Along Track Error (ATE) and Cross Track Error (CTE), which evaluate the longitudinal and lateral deviations with respect to the ground-truth trajectory. CTE is much more impactful for driving as it can cause the autonomy system to confuse which lanes are occupied and thus cause the SDV to suddenly brake putting itself and nearby drivers at risk. We show that our method largely outperforms the baselines on all metrics In particular, we see very large gains on BrierCTE, indicating that our method has a much better map understanding. Due to the high speeds on the highway, small lateral deviations from the centerlines compound over time, causing large cross-track errors at 6s for the baselines. For context, a truck only has around 30 cm buffer on each side of the lane taking into account its mirrors, so those errors can be critical. Similar to results in Table I, the BrierMinFDE metric shows our better calibrated mode probabilities.

**Importance of the heterogeneous scene encoder:** $M_1$ in Table III entirely bypasses this component, essentially feeding agent and map features to the decoder directly out of the agent history and lane-graph encoders. We see $6\%$ higher BrierMinFDE@K=6.

**Importance of goal-based decoder:** $M_2$ uses a simple MLP instead of our heterogeneous GNN to predict the trajectory mixture. This is the component that has the biggest impact, increasing the BrierMinFDE@K=6 by $51\%$ when replaced.

**Importance of greedy goal sampler:** $M_3$ replaces the proposed greedy goal sampler with top-k, an approach used by [6]. We see $14\%$ higher BrierMinFDE@K=6.

**Importance of pair-wise relative heterogeneous GNN:** $M_4$ replaces this component with a simpler GNN without edge attributes. Geometric information is now encoded on the nodes attributes in a global coordinate frame as proposed in [4], [5], [9]. We see $30\%$ higher BrierMinFDE@K=6.

**Qualitative results:** We visualize GoRela's predictions in a diverse set of scenarios from Argoverse 2 in Fig. 4. The first 3 columns show scenarios in which our model accurately predicts a mode very close to the ground-truth for the focal agent, which we emphasize are mined for hard cases. From the visualizations, it is clear our model achieves a good understanding of the map as well as the agent-agent interactions, predicting realistic multi-modal distributions. The last column showcases failure modes of our model such as predicting an agent will lane change when in fact is keeping its lane, and cutting corners too much in a left turn.

## VI. Conclusion

In this work, we have proposed GoRela, a motion forecasting model that predicts more accurate multi-agent multi-modal trajectories both in urban and highway roads. As shown by our experiments, our model achieves this through several contributions: (i) a viewpoint-invariant architecture that makes learning more efficient thanks to the proposed pair-wise relative positional encoding, (ii) a versatile graph neural network that understands interactions in heterogeneous spatial graphs with agent and map nodes related by multi-edge adjacency matrices (e.g., lane-lane, agent-lane, agent-agent), and (iii) a probabilistic goal-based decoder that leverages the lane-graph to propose realistic goals paired up with a simple greedy sampler that encourages diversity.

[1] https://bit.ly/argo2_eval_metrics

## REFERENCES


[1] F. Janjoš, M. Dolgov, and J. M. Zöllner, "Starnet: Joint action-space prediction with star graphs and implicit global-frame self-attention," in *2022 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2022, pp. 280–286. 1, 2

[2] J. Gao, C. Sun, H. Zhao, Y. Shen, D. Anguelov, C. Li, and C. Schmid, "Vectornet: Encoding hd maps and agent dynamics from vectorized representation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 525–11 533. 1, 2

[3] H. Cui, V. Radosavljevic, F.-C. Chou, T.-H. Lin, T. Nguyen, T.-K. Huang, J. Schneider, and N. Djuric, "Multimodal trajectory predictions for autonomous driving using deep convolutional networks," *arXiv preprint arXiv:1809.10732*, 2018. 1, 2, 5

[4] J. Ngiam, V. Vasudevan, B. Caine, Z. Zhang, H.-T. L. Chiang, J. Ling, R. Roelofs, A. Bewley, C. Liu, A. Venugopal, *et al.*, "Scene transformer: A unified architecture for predicting future trajectories of multiple agents," in *International Conference on Learning Representations*, 2021. 1, 2, 3, 5, 6

[5] T. Gilles, S. Sabatini, D. Tsishkou, B. Stanciulescu, and F. Moutarde, "Thomas: Trajectory heatmap output with learned multi-agent sampling," *arXiv preprint arXiv:2110.06607*, 2021. 1, 2, 3, 5, 6

[6] W. Zeng, M. Liang, R. Liao, and R. Urtasun, "Lanercnn: Distributed representations for graph-centric motion forecasting," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 532–539. 1, 2, 4, 6

[7] Y. Chai, B. Sapp, M. Bansal, and D. Anguelov, "Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction," *arXiv preprint arXiv:1910.05449*, 2019. 2, 5

[8] M. Bansal, A. Krizhevsky, and A. Ogale, "Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst," *arXiv preprint arXiv:1812.03079*, 2018. 2

[9] M. Liang, B. Yang, R. Hu, Y. Chen, R. Liao, S. Feng, and R. Urtasun, "Learning lane graph representations for motion forecasting," in *ECCV*, 2020. 2, 3, 4, 5, 6

[10] S. Casas, W. Luo, and R. Urtasun, "Intentnet: Learning to predict intention from raw sensor data," in *Conference on Robot Learning*, 2018. 2

[11] J. Hong, B. Sapp, and J. Philbin, "Rules of the road: Predicting driving behavior with a convolutional model of semantic interactions," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[12] F.-C. Chou, T.-H. Lin, H. , V. Radosavljevic, T. Nguyen, T.-K. Huang, M. Niedoba, J. Schneider, and N. Djuric, "Predicting Motion of Vulnerable Road Users using High-Definition Maps and Efficient ConvNets," *arXiv e-prints*, Jun 2019. 2

[13] W. Zeng, W. Luo, S. Suo, A. Sadat, B. Yang, S. Casas, and R. Urtasun, "End-to-end interpretable neural motion planner," in *Proceedings of the IEEE CVPR*, 2019. 2

[14] S. Casas, C. Gulino, R. Liao, and R. Urtasun, "Spagnn: Spatially-aware graph neural networks for relational behavior forecasting from sensor data," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 9491–9497. 2

[15] S. Casas, C. Gulino, S. Suo, K. Luo, R. Liao, and R. Urtasun, "Implicit latent variable model for scene-consistent motion forecasting," *ECCV 2020*, 2020. 2

[16] A. Cui, S. Casas, A. Sadat, R. Liao, and R. Urtasun, "Lookout: Diverse multi-future prediction and planning for self-driving," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 16 107–16 116. 2

[17] S. Casas, A. Sadat, and R. Urtasun, "Mp3: A unified model to map, perceive, predict and plan," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 14 403–14 412. 2

[18] W. Zeng, S. Wang, R. Liao, Y. Chen, B. Yang, and R. Urtasun, "Dsdnet: Deep structured self-driving network," in *ECCV*, 2020. 2

[19] L. Li, B. Yang, M. Liang, W. Zeng, M. Ren, S. Segal, and R. Urtasun, "End-to-end contextual perception and prediction with interaction transformer," *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020. 2

[20] B. Varadarajan, A. Hefny, A. Srivastava, K. S. Refaat, N. Nayakanti, A. Cornman, K. Chen, B. Douillard, C. P. Lam, D. Anguelov, *et al.*, "Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 7814–7821. 2

[21] T. Gilles, S. Sabatini, D. Tsishkou, B. Stanciulescu, and F. Moutarde, "Gohome: Graph-oriented heatmap output for future motion estimation," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 9107–9114. 2

[22] X. Mo, Z. Huang, Y. Xing, and C. Lv, "Multi-agent trajectory prediction with heterogeneous edge-enhanced graph attention network," *IEEE Transactions on Intelligent Transportation Systems*, 2022. 2, 4

[23] S. Casas, C. Gulino, R. Liao, and R. Urtasun, "Spagnn: Spatially-aware graph neural networks for relational behavior forecasting from sensor data," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020. 2

[24] J. Wang, T. Ye, Z. Gu, and J. Chen, "Ltp: Lane-based trajectory prediction for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 17 134–17 142. 2

[25] X. Jia, P. Wu, L. Chen, H. Li, Y. Liu, and J. Yan, "Hdgt: Heterogeneous driving graph transformer for multi-agent trajectory prediction via scene encoding," 2022. 2

[26] S. Casas, C. Gulino, S. Suo, and R. Urtasun, "The importance of prior knowledge in precise multimodal prediction," *arXiv preprint arXiv:2006.02636*, 2020. 2

[27] N. Rhinehart, K. M. Kitani, and P. Vernaza, "R2p2: A reparameterized pushforward policy for diverse, precise generative path forecasting," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 772–788. 2

[28] B. Ivanovic and M. Pavone, "The trajectron: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 2375–2384. 2

[29] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 2011, pp. 627–635. 2

[30] H. Zhao, J. Gao, T. Lan, C. Sun, B. Sapp, B. Varadarajan, Y. Shen, Y. Shen, Y. Chai, C. Schmid, *et al.*, "Tnt: Target-driven trajectory prediction," *arXiv preprint arXiv:2008.08294*, 2020. 2

[31] J. Gu, C. Sun, and H. Zhao, "Densetnt: End-to-end trajectory prediction from dense goal sets," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 15 303–15 312. 2

[32] T. Gilles, S. Sabatini, D. Tsishkou, B. Stanciulescu, and F. Moutarde, "Home: Heatmap output for future motion estimation," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2021, pp. 500–507. 2

[33] N. Deo and M. M. Trivedi, "Trajectory forecasts in unknown environments conditioned on grid-based plans," *arXiv preprint arXiv:2001.00735*, 2020. 2

[34] N. Deo, E. Wolff, and O. Beijbom, "Multimodal trajectory prediction conditioned on lane-graph traversals," in *Conference on Robot Learning*. PMLR, 2022, pp. 203–212. 2

[35] Y. Wang, H. Zhou, Z. Zhang, C. Feng, H. Lin, C. Gao, Y. Tang, Z. Zhao, S. Zhang, J. Guo, *et al.*, "Tenet: Transformer encoding network for effective temporal flow on motion prediction," *arXiv preprint arXiv:2207.00170*, 2022. 2

[36] N. Nayakanti, R. Al-Rfou, A. Zhou, K. Goel, K. S. Refaat, and B. Sapp, "Wayformer: Motion forecasting via simple & efficient attention networks," *arXiv preprint arXiv:2207.05844*, 2022. 2

[37] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017. 3

[38] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *stat*, vol. 1050, p. 20, 2017. 3

[39] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988. 5

[40] B. Wilson, W. Qi, T. Agarwal, J. Lambert, J. Singh, S. Khandelwal, B. Pan, R. Kumar, A. Hartnett, J. K. Pontes, *et al.*, "Argoverse 2: Next generation datasets for self-driving perception and forecasting," in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. 5

[41] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *International conference on machine learning*. PMLR, 2017, pp. 1321–1330. 6