StarNet: Joint Action-Space Prediction with Star Graphs and Implicit Global Frame Self-Attention

Faris Janjoš¹, Maxim Dolgov¹, and J. Marius Zöllner²

Abstract—In this work, we present a novel multi-modal multi-agent trajectory prediction architecture, focusing on map and interaction modeling using graph representation. For the purposes of map modeling, we capture rich topological structure into vector-based star graphs, which enable an agent to directly attend to relevant regions along polylines that are used to represent the map. We denote this architecture StarNet, and integrate it in a single-agent prediction setting. As the main result, we extend this architecture to joint scene-level prediction, which produces multiple agents’ predictions simultaneously. The key idea in joint-StarNet is integrating the awareness of one agent in its own reference frame with how it is perceived from the points of view of other agents. We achieve this via masked self-attention. Both proposed architectures are built on top of the action-space prediction framework introduced in our previous work, which ensures kinematically feasible trajectory predictions. We evaluate the methods on the interaction-rich inD and INTERACTION datasets, with both StarNet and joint-StarNet achieving improvements over state of the art.

I. INTRODUCTION

Accurate prediction of the driving situation is a major cornerstone for achieving performant full autonomy of self-driving cars. Despite a strong research and industry focus, there are many problems to be solved, such as understanding complex social interactions among different agents and effectively incorporating rich topological information. Other important aspects are prediction of multi-modal trajectories, conditioning predictions on assumed goals of given agents, as well as achieving reasonable long-term predictions. In tackling these challenges, Deep Neural Networks (DNN) have shown great results over classical robotics approaches, especially for the use-case of urban driving.

The challenges of environment representation and interaction modeling are tightly coupled, e.g. the maneuvers of two negotiating vehicles in a highly-interactive situation are constrained by the road topology. Therefore, a learned model must consider the effects of topology on the driving situation. This makes the representation of map information paramount: it should enable explicitly relating to map elements such as lane centerlines and road boundaries, as well as segments along these elements. Furthermore, it must allow the model to discern between more and less important

1 Robert Bosch GmbH, Corporate Research, Advanced Autonomous Systems, 71272 Renningen, Germany. {faris.janjos, maxim.dolgov}@de.bosch.com
2 Research Center for Information Technology (FZI), 76131 Karlsruhe, Germany, zoellner@fzi.de

This work was financially supported by the Federal Ministry of Economic Affairs and Energy of Germany, grant number 19A20026H, based on a decision of the German Bundestag.

Fig. 1: Example of a highly interactive situation requiring joint scene prediction: vehicle 1 slows down to allow vehicle 2 to overtake parked vehicle 4; vehicle 3 must reverse behind vehicle 4.
• A novel, star-graph-of-vectors polyline representation, with unconstrained field-of-view (given a map description) and direct modeling of most relevant segments.
• A mechanism to explicitly combine features from multiple reference frames via self-attention while masking out irrelevant information, enabling joint scene-level prediction.
• State-of-the-art performance on the INTERACTION dataset, containing challenging roundabout, intersection, and highway merge scenarios.

II. RELATED WORK

The field of trajectory prediction has generated an exhaustive literature. Within the field of autonomous driving, the multifaceted nature of the problem yields works that focus on specific challenges within the overall landscape. They include but are not limited to: environment representation, multi-agent interaction, multi-modality, goal-conditioning, and kinematic constraints. Furthermore, some works integrate prediction with detection and planning, and extract a larger receptive field. Consequently, they are able to capture a longer range longitudinally, as well as account for the different semantic meaning of lateral lanes. Another class of works is and , which model attention operation with additional adjacency matrices in order to determine the most relevant vectors without artificially limiting the receptive field. Furthermore, our map representation is simpler to pre-process since we use standard graph convolutions, as well as , since we do not manually select the reference polyline and allow other map element types to be attended to as well.

A. Graph-based map representation

Encoding map information into graphs and using GNNs offers several advantages over image-based inputs used in conjunction with CNNs. CNNs extract locational features from rasterized BEV images or grids with LiDAR point clouds. In contrast, GNNs can learn directly from graph-based representations, which encode the geometric structure of the road into nodes and edges. In doing so, they alleviate the need to infer objects from pixels, as well as improve efficiency due to fewer weights. Furthermore, graphs benefit from a larger field of view than rasters, which are usually restricted by image dimensions and resolution. These factors contribute to a substantially higher representation density.

VectorNet is a seminal work that uses graph-based map representations. This approach fits polylines to map elements and dissects them into their constituent vectors. Then, fully-connected graphs are constructed per map element, which are aggregated by a GNN into a feature vector. This procedure is used as a basis in further works. Alternative approaches are and , redefining the graph convolution operation with additional adjacency matrices in order to capture a larger receptive field. Consequently, they are able to capture a longer range longitudinally, as well as account for the different semantic meaning of lateral lanes. Another class of works is and , which model attention to specific lanes or sections along a reference polyline, constructed by concatenating individual polyline points. In the case of , this enables placement of hypothetical goals along the polyline to condition the trajectory prediction.

Works like VectorNet construct fully-connected subgraphs for each map element in order to mitigate the GNN information bottleneck problem. This enables each graph node to be no more than one hop away from any other node. However, as a result, unnecessary information is shared between vectors that are not physically close but are part of the same polyline. Thus, only considers the nodes within a distance threshold to the predicted vehicle, in turn limiting the receptive field in aggregation. Furthermore, includes ordering information into the node attributes via an integer index, which raises correctness questions since integers are combined with floating point features such as positions.

Regarding the aspect of our work, instead of connecting vectors by their polyline membership, we ask the question, which parts of a polyline are the most relevant to an agent? We task a Graph Attention Network (GAT) with the answer; the attention mechanism learns to determine the most relevant vectors without artificially limiting the receptive field. Furthermore, our map representation is simpler to pre-process since we use standard graph convolutions, as well as , since we do not manually select the reference polyline and allow other map element types to be attended to as well.

B. Joint graph-based interaction modeling

Social interaction in a driving scene can inherently be represented as a natural graph, where nodes are agents and edges model their (weighted) connections. Hence, virtually all recent state-of-the-art approaches use graph-based learned models such as GNNs and Multi-Head Attention (MHA), which is related to the Transformer architecture.

For the sake of brevity, we limit our review to joint prediction works, which is the randomness of their outputs because they require sampling from a latent distribution during inference. Thus, likelihood estimation of the predicted trajectory distribution is difficult. This is exacerbated in the single-step prediction approaches, which construct a trajectory iteratively. Another drawback is limited map information; approaches either don’t consider the map at all or use global representations centered around the AV. As mentioned in Sec. the AV-centered global view requires extracting local patches around each agent of interest and relating information implicitly via CNNs. As an alternative, and perform joint prediction in the global reference frame without AV-centering, but reduce the effects of arbitrary

1 Modeling the full map connectivity as a mesh-like natural graph and interweaving the agents nodes would result in long chains of propagated information. Learning over such a graph would necessitate many iterations of message passing and yield over-smoothed node embeddings.

2 Incidentally, the basic Transformer layer is equivalent to the GNN GAT layer, in the case of multiple attention heads and a fully-connected underlying graph.
origin placement by injecting the global-frame map via cross-attention and Long Short-Term Memory (LSTM)-like implicit gating, respectively.

Regarding the joint interaction modeling aspect of our work, we start with deterministic one-shot prediction outputs given deterministic inputs. We model the map explicitly from local-frame graph representations with unlimited field of view. Our work is closest to the prediction model in [20], however, instead of using global reference frames, we ask the question, how is an agent jointly perceived by other agents? We arrive at a MHA model with masking, which combines multiple local representations to construct an implicit global frame. Furthermore, our model is smaller since it uses two GAT and two MHA layers to model the map and social interaction, compared to 18 MHA layers in [20]. Finally, we frame the model in the action-space framework of our previous work [23], ensuring kinematically feasible predictions.

III. METHOD

A. Background

Consider the task of vehicle trajectory prediction in a driving situation with $N$ heterogeneous interacting agents. We define the single-agent prediction problem as predicting the distribution of future waypoints $\hat{Y}_i$ of vehicle $i$. It can be framed in an imitation learning setting, where a learned model parameterizes the distribution

$$\hat{Y}_i \sim P(\hat{Y}_i|D^i) \ ,$$

conditioned on the local context $D^i$ of vehicle $i$. Here, the superscript indicates that the values are represented in the local reference frame of vehicle $i$, subscript indicates prediction for agent $i$, while $\hat{\cdot}$ denotes predicted future values. We simplify the problem in (1) by predicting a sample $\hat{Y}_i$ of the distribution, e.g. a $2 \times T$ matrix of $xy$ coordinates over $T$ future time steps.

The context $D^i = \{M^i, T^i\}$ contains the map $M^i$ and past position tracks $T^i$ of vehicle $i$ and its neighboring $N - 1$ agents, with $T^i = \{X^i_j\}_{j=1}^N$. Each track $X^i_j$ is a $3 \times T$ matrix of $xy$ coordinates of agent $j$ over $T$ past time steps (in the reference frame of vehicle $i$ at the current time step) and a padding row, since the agent might not be present in the scene for each time step. The padding row contains zeros for non-existent points (and zero positions) and ones otherwise.

In joint prediction, we consider the task of predicting $K$ ($K \leq N$) vehicles’ trajectories simultaneously. Thus, the learned model parameterizes the distribution

$$\hat{Y} \sim P(\hat{Y}|D) \ .$$

Therefore, we predict a sample $\hat{Y}$ of $\hat{Y}$ for future trajectories of all $K$ vehicles, given their contexts $D$, where $\hat{Y} = \{Y_k\}_{k=1}^K$ and $D = \{D^k\}_{k=1}^K$, respectively. Each $D^k$ can be separated into its map and tracks components, $M^k$ and $T^k = \{X^k_j\}_{j=1}^N$. Note that tracks $T^k$ contain trajectories for all $N$ agents, including those whose waypoints are not predicted such as pedestrians and bicycles.

In parameterizing the distributions in (1) and (2), we use a general encoder–decoder structure, with a context encoder and an output decoder. The encoder reasons about the context $D$ while the decoder generates predicted trajectories $\hat{Y}$. Furthermore, we make use of track encoders, e.g. we encode each agent track $X^i_j$ ($i \in [1, ..., K], j \in [1, ..., N]$) into a feature vector $z^i_j$ via a 1D CNN network.

B. Action-space prediction

We use positional representations in modeling the environment $D$ within the context encoder. Both the polyline map $M$ (as will be shown later), as well as the tracks $T$, contain $xy$ coordinates describing polyline control points and past trajectories, respectively. Hence, in generating learned feature representations of the environment, map-dependent interactions are inferred from data in the Euclidean space.

However, when generating the predicted trajectories $\hat{Y}$ via the decoder, we shift the learning problem into the action-space of accelerations and steering angles. We provide past actions as action tracks $A^i_j$ ($i \in [1, ..., K]$) and generate future actions with a Gated Recurrent Unit (GRU) [37]. This is consistent with the action-space prediction framework [23] and guarantees that a learned model does not have to capture motion models, as well as ensuring kinematic feasibility (with an output kinematic model). Similarly to position tracks, we encode past action tracks $A^i_j$ into feature vectors $w^i_j$ with the same network.

C. Single-agent prediction with StarNet

In this section, we present a single-agent prediction method for regressing the future trajectory of vehicle $i$ (prediction-ego). We approximate (1) via a deterministic encoder–decoder model. The encoder consists of a star graph map model and a map-dependent interaction model, while the decoder is a multi-modal action predictor, directly generating $m$ samples from the distribution (1).

1) Star graph map model: Key component of single-agent StarNet is its representation of map elements via star graphs. First, each map element, such as a sidewalk, lane center-line, or a traffic island, is approximated by a polyline consisting of fixed-length vectors, similarly as in VectorNet [9]. Thus, the representation $M^i$ of the map in the agent $i$’s reference frame consists of $Q$ polylines

$$M^i = \{q^i_j\}_{j=1}^Q \ .$$

In turn, each polyline consists of $L$ vectors comprising their start and end $xy$ coordinates and one-hot type encoding

$$q^i_j = \{v^i_j\}_{l=1}^L \ ,$$

$$v^i_{jl} = [v_{start}, v_{end}, v_{type}]^T \ .$$

Given this polyline representation, we construct a directed star graph for each polyline $q^i_j$. In this structure, the past prediction-ego track is the central node with embedded track features $z^i_j$ and edges connecting each vector $v^i_{jl}$ to the central node. To ensure message passing compatibility, we embed the vector nodes (5) into the same dimensionality as the features $z^i_j$ using a linear layer. Finally, we feed this graph to a 2-layer GAT and aggregate the nodes via max-pooling to obtain polyline-level embeddings $q^i_j$ of same dimensionality as the nodes. This structure is depicted in Fig. [2].
Fig. 2: StarNet single-agent architecture for the driving scene in Fig. 1. Vehicle 1 is the prediction-ego, and two polylines and three neighboring agents are in its local context. First, the two polylines determined by five vectors each are represented in a directed star graph with prediction-ego track embedding $z_1^i$ in the center and vector embeddings $v_{1,2,5}^i$ outward. Polyline-level embeddings are generated by a 2-layer GAT, followed by a 1-layer Multi-Head Attention modeling map-dependent social interaction via aggregating all agents’ and polylines’ embeddings, $z_{1,4}^i$ and $q_{1,2}^i$ respectively. Then, prediction-ego embedding is selected, concatenated with its action track embedding $w_1^i$, and fed through an action-prediction GRU to predict future actions $\hat{a}_1^i$. Finally, positions $\hat{x}_1^i$ are obtained via a kinematic model transformation.

The star graph and the accompanying GAT model the relationship between the ego track and the map. Contrary to VectorNet’s fully-connected graphs, we assume that there is more information contained in the vehicle’s direct attention (represented by its past track embedding) to a specific vector within a polyline rather than between the polyline vectors themselves. This allows us to expand the receptive field, since the attention mechanism will learn to ignore distant vectors, and to consider them proportionally to their weights in aggregation. Furthermore, the structure removes the need to include artificial ordering into the vector (5), as it is done in VectorNet in order to help convey the polyline geometry, which simplifies the learning problem.

2) Map-dependent interaction model: In the single-agent StarNet, we model map-dependent social interaction with a MHA [6], see Fig. 2. We combine vehicle $i$’s past track embedding $z_1^i$ with polyline embeddings $q_j^i$ ($j \in \{1, ..., Q\}$), as well as track embeddings $z_j^i$ ($j \in \{1, ..., N - 1\}$) of each agent sharing the scene with $i$. Then, we stack the embedding vectors into a matrix with $N + Q$ rows and feed it into a single MHA layer. Here, linear projections of the input are generated in the form of query, key, and value matrices. Then, the self-attention operation [6] is applied in order to infer the relationships between the embeddings. The output of the MHA is of the same dimensionality as the stacked input matrix, and we select the row that corresponds to the vehicle $i$. Through the GAT and MHA layers, the obtained embedding is able to capture the map-dependent social interaction in the local context of the prediction-ego. The next step is feeding it into the action output decoder.

3) Multi-modal action decoder: The action decoder combines the positional embedding of the prediction-ego, aggregated to consider the map-dependent social interaction, with its action embedding. We concatenate $z_1^i$ with $w_1^i$ and feed it into the action decoder, which is the same GRU network as in [23] (depicted in Fig. 2). It generates steering angles and accelerations, directly predicting $m$ action modes (action
trajectories and softmax scores, which can be interpreted as probabilities). The modes are converted to predicted vehicle positions via a bicycle kinematic model, fully capturing kinematic characteristics of motion.

We train the whole pipeline with the loss

$$\mathcal{L} = \mathcal{L}_{\text{reg}} + \beta \mathcal{L}_{\text{class}},$$

where $\mathcal{L}_{\text{reg}}$ considers the mismatch to the ground-truth future trajectory and $\mathcal{L}_{\text{class}}$ considers the mode probability via cross-entropy, same as in [23], with $\beta$ set to 1.

### D. Joint prediction with joint-StarNet

In this section, we present a joint prediction method for regressing the future trajectory of $k$ vehicles in a driving scene ($k \in [1, ..., K], K \leq N$). We directly model the joint distribution $\mathbb{P}(\mathbf{x}_t | \mathbf{X}_{<t}, \mathbf{X}_{>t})$ without factorizing individual agents. The joint-StarNet architecture builds on StarNet from Sec. III-C and achieves joint prediction by aggregating local features into an implicit global frame via masked self-attention.

1) Implicit global frame self-attention: The joint-StarNet is an extension to single-agent StarNet. After determining the joint prediction candidates, we perform the first two steps of the single-agent StarNet pipeline separately for each vehicle. We construct local map element star graphs, aggregate them with GATs and combine them with locally embedded tracks in the Single-Agent MHA blocks. Then, we select the positional embeddings corresponding to each of the joint agents, in each joint agent’s local context, obtaining $K^2$ feature vectors $\{z_j^k\}_{j=1}^K \in \mathbb{R}^{K \times 1}$. An example is provided in Fig. 3a. This combination of features contains mutual local information about each joint prediction candidate, at the cost of quadratically increasing number of features. Nevertheless, we do not observe a computational bottleneck due to efficient batching in training, described in Sec. IV-A.

Given the individual local features, we now construct an implicit global frame by combining features from each local frame. We achieve this with another MHA block (denoted as Joint Multi-Head Attention in Fig. 3a), taking in features $\{z_j^k\}_{j=1}^K \in \mathbb{R}^{K \times 1}$ stacked into a matrix. In the output, we select the rows corresponding to features $\{z_k^j\}_{k=1}^K$ (in their respective reference frames) and feed them in a batched manner into an action decoder block. We can train with the same loss (6) as in single-agent StarNet training.

2) Attention-mask: In combining multiple local contexts into an implicit global context, the embeddings corresponding to a single vehicle in different local frames should attend only to themselves. We achieve this by limiting the self-attention with a $K^2 \times K^2$ attention mask matrix. It ensures that only the features $\{z_j^k\}_{k=1}^K$ for the agent $j$ in different frames are considered, in each row of the stacked input matrix. This is exemplified in Fig. 3b.

The joint-StarNet architecture with masking allows to explicitly combine multiple local interaction models and integrate them into an implicit global interaction model while accounting for non-symmetric attention. Each local, single-agent model uses direct map representations that condition the local social interaction.

---

**TABLE I:** Comparison of StarNet and joint-StarNet with approaches in literature (reported results). The values for [59] and [15] are given in [16]. Since [34] reports results for different map types separately, we computed the aggregate value by combining the ratios of specific map types in the validation dataset.

| Method       | ADE   | FDE   |
|--------------|-------|-------|
| FFW-ASP [23] | 0.37  | 1.02  |
| VectorNet [9] | 0.37  | 1.03  |
| StarNet      | 0.35  | 0.97  |
| joint-StarNet | 0.36  | 1.02  |
| joint-StarNet | 0.32  | 0.89  |

**TABLE II:** Comparison of StarNet and joint-StarNet with approaches in literature (reported results). The values for [59] and [15] are given in [16]. Since [34] reports results for different map types separately, we computed the aggregate value by combining the ratios of specific map types in the validation dataset.

| Method       | ADE   | FDE   |
|--------------|-------|-------|
| DESIRE [59]  | 0.32  | 0.88  |
| MultiPath [15] | 0.30  | 0.99  |
| STG-DAT [15] | 0.29  | 0.54  |
| TNT [10]     | 0.21  | 0.67  |
| ReCoR [40]   | 0.19  | 0.66  |
| HEAT-I-R [35] | 0.19  | 0.65  |
| ITRA [8]     | 0.17  | 0.49  |
| StarNet      | 0.16  | 0.49  |
| joint-StarNet | 0.13  | 0.38  |

---

**IV. RESULTS**

### A. Implementation

For implementing the StarNet (Sec. III-C) and joint-StarNet (Sec. III-D) architectures we used several network types: 1D CNNs, GATs, MHA, and GRUs. The track encoder 1D CNNs are adapted from the ActorNet model in [10] and embed position and action tracks to 128- and 64-dimensional vectors, respectively. The position embeddings are then used as nodes in the GAT and MHA networks, which are both realized with 8 attention heads. In the action decoder block, the concatenated position and action track embeddings are first transformed by two linear layers of sizes \{512, 256\} (with batch normalization and tanh activation) before being fed into the GRU network. The GRU iterates these transformed features three times through two layers of 512 hidden units, directly predicting $m$ future action modes.

The joint-StarNet has several important practical considerations. Within the model, each joint candidate is first fed through the first two steps of the single-agent StarNet (Sec. III-C and Sec. III-C.2), and then the features from multiple local contexts are aggregated in the Joint MHA, as exemplified in Fig. 3a. In a single batch element, this induces linearly growing complexity in the GAT and Single-agent MHA and quadratically growing complexity in the Joint MHA, with the number of joint candidates. However, the computational load does not grow equally due to efficient batching of different scenes. In the GAT case, we aggregate different star graphs into a single graph with a block-diagonal adjacency matrix. Similarly, in both MHA blocks we feed inputs from different scenes together in a batch, but use additional batch-wise attention mask. As a result, this brings a higher utilization of GPU memory. However, reasonably-sized batches are made possible by the compact input representations.
ment outside an \( \epsilon \) by \( 0 \)
(modal) and did not incorporate the self-supervised map
modal action output decoder in VectorNet (originally uni-
tecture from our previous work \cite{23} and our own im-
D. Performance

We compared our approaches against the raster-based
Feed-Forward Action-Space Prediction (FFW-ASP) archi-
ture from our previous work \cite{23} and our own im-
mentation of VectorNet \cite{9}, with the results shown in
Tab. \ref{tab:results}. To ensure a fair comparison, we used our multi-
modality within the action output decoder, which does not condition predicted modes on other vehicles’ predicted
modes. Furthermore, we plan to integrate the presented
architectures with the self-supervised long-term prediction
framework of \cite{23}, which predicts future context representa-
tions prior to trajectories. We expect that the denser map
and joint interaction modeling will lead to improved context
prediction, bringing further performance improvements.

\section{Conclusion}

In this work, we presented an attention-based approach
to directly represent map elements and explicitly model
mutual social interaction. We offered two novel architectures,
the single-agent StarNet that models map elements as star
graphs, and a joint prediction extension via an additional
MHA layer. The joint-StarNet can handle a variable number
of agents and integrate their local awarenesses into an im-
licit global model. In this sense, it contributes an important
step towards joint scene understanding.

In future work, we will focus on the multi-modality aspect
of joint prediction and address the shortcomings mentioned
in Sec. \ref{sec:future_work}. We plan to improve on the implicit modeling of
multi-modality within the action output decoder, which does not condition predicted modes on other vehicles’ predicted
modes. Furthermore, we plan to integrate the presented
architectures with the self-supervised long-term prediction
framework of \cite{23}, which predicts future context representa-
tions prior to trajectories. We expect that the denser map
and joint interaction modeling will lead to improved context
prediction, bringing further performance improvements.
