Improving Multi-Agent Motion Prediction with Heuristic Goals and Motion Refinement

Carlos Gómez-Huélamo[1]*, Marcos V. Conde[2], Rafael Barea[1], Luis M. Bergasa[1]
[1]Universidad de Alcalá, Spain [2]University of Würzburg, Germany
{carlos.gomez, rafael.barea, luism.bergasa}@uah.es
marcos.conde@uni-wuerzburg.de

Abstract

Motion Prediction (MP) of multiple surrounding agents in physical environments, and accurate trajectory forecasting, is a crucial task for Autonomous Driving Stacks (ADS) and robots. Current methods for MP use end-to-end pipelines, where the input data is usually a HD map and the past trajectories of the most relevant agents; leveraging this information is a must to obtain optimal performance. In that sense, a reliable Autonomous Driving (AD) system must produce fast and accurate predictions to ensure traffic safety.

In this work, we tackle Multi-Agent Motion Prediction using an end-to-end pipeline that combines Deep Learning (DL) and heuristic scene understanding. Our model uses as input the map of the scene, the past trajectories of the agents, and additional information about the scene geometry and agents e.g., type of agent, lane distribution.

We design our model using powerful attention mechanisms with GNNs to enhance agents interactions, heuristic proposals as preliminary plausible information and a motion refinement module to further improve temporal consistency. We achieve SOTA results on the Argoverse 2 Motion Forecasting Benchmark reducing in millions of parameters previous methods such as GANet, and improving over LaneGCN. Our code is available at https://github.com/Cram3r95/argo2goalmp.

1. Introduction

Autonomous Driving (AD) is a trendy research topics in academia and industry due to its real-world impact. Assuming the surrounding agents have been detected and tracked *i.e.* we have their past trajectories during a time interval, the core task of the perception layer is Motion Prediction (MP), that is, predicting the future trajectories [2–4] of the surrounding traffic agents in the environment -also known as actors- given the past on-board sensor and map information, taking into account the corresponding traffic rules, the scene, and social interaction among the agents. This is essential for the AV to make safe and reasonable decisions in the subsequent planning and control module.

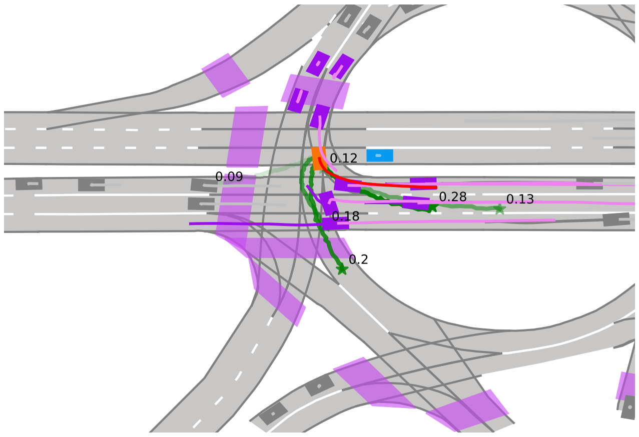

Figure 1. Complex roundabout scenario in Argoverse 2 [1] Motion Forecasting dataset. We represent: our vehicle (**ego**), the **focal agent**, the **relevant agents** in the scene, and **other agents**. We can also see the **ground-truth** trajectory of the target agent, our **multimodal predictions** (with the corresponding **confidences**). We also highlight the most important topology of the road, such as pedestrian crossing and boundaries mark type.

These predictions are typically multi-modal *i.e.* given the past motion of a particular vehicle and its surrounding scene, there may exist more than one possible future behaviour -also known as modes-. Therefore, MP models need to cover the different choices that a driver could make (*e.g.* going straight, turning, accelerate) as a possible discrete trajectory in the immediate future, or in a probabilistic manner [5, 6] (*e.g.* potential area of movement heatmaps).

Traditional methods for motion forecasting [7–9] are based on physical kinematic constraints and road map information with heuristics. These approaches fail to capture

*Corresponding author

the rich behavior strategies and interaction in complex scenarios, in such a way they are only suitable for simple prediction scenes and short-time prediction tasks [10].

Moreover, the advances in Deep Learning (DL) and the recent emergence of large-scale datasets with high-definition maps (HD maps) allow us to understand and capture the complexity of a driving scenario using data-driven methods [11–13] and achieve the most promising *state-of-the-art* results by learning such intrinsic rules, and agent interactions. Thus, the core challenge for MP resides in how to effectively compute and integrate the surrounding environment, both in terms of agents and map information, to predict multiple reliable trajectories.

Methods based on Graph Neural Networks (GNNs) [14] [15] have achieved SOTA results on the most relevant benchmarks for Motion Prediction, though these methods might lack interpretability and control regarding the graph-based modules [16]. In this work we focus on attention **transformer-based** approaches such as [16–18].

In these DL models, an encoder usually takes into account: (i) multiple-agents **history** states (position, velocity, etc.), and (ii) a High Definition **(HD) Map** [19] that includes: intersections, traffic lights and signals, multi-channel codification or complex vectorization [5, 15]. Note that obtaining and fusing this information (*e.g.* actor-to-actor, map-to-actor) is a research topic by itself [15, 20, 21] and a core part in the AD pipeline. Here we identify a bottleneck for efficient real-time applications [7, 8], as usually, more (complex) data-inputs implies higher model complexity and inference time [22].

Predicting the future trajectories of the agents without considering their nature might not be optimal (*e.g.* predicting on a pedestrian, a cyclist or a car using the same logic). For this reason, we integrate additional features related to the type and properties of agents. Moreover, we also include heuristic scene understanding to constrain the model predictions towards the real scene geometry (*e.g.* plausible centerlines and lanes).

Most *state-of-the-art* methods require an overwhelmed amount of information as input, specially in terms of the physical context -HD maps-; this usually implies more model complexity, and these can be inefficient in terms of computation [19, 22, 23]. However, we reduced notably the complexity of our model *w.r.t* previous methods such as GANet [24] to avoid these possible contraints.

In this paper, we aim to achieve accurate trajectory forecasting, yet, using light-weight transformer-based models. We make the following **contributions**:

1. We present a SOTA method on the Argoverse 2 Motion Forecasting Benchmark.

2. Our model uses various attention mechanisms with GNNs, and a motion refinement module to further improve temporal consistency.

3. In comparison to previous methods that rely only on past trajectories and HD map, we additionally use information about the agents (*e.g.* type of agent) and the scene geometry (*e.g.* lane distribution and possible goal points).

4. Our method reduces in millions of parameters previous methods such as GANet [24], and improves over LaneGCN [15].

5. Finally, we provide an open-source framework for MP.

2. Related Work

One of the crucial tasks that Autonomous Vehicles (AV) must face during navigation, specially in arbitrarily complex urban scenarios, is to predict the behaviour of dynamic obstacles [4, 25–27].

Traditional methods [10, 25] usually consider only physics-related factors (*e.g.* the velocity and acceleration of the target vehicle) and road-related factors (predictions must be in the proper lane), and are only suitable for *short-time* prediction tasks [10] and simple traffic scenarios, *e.g.* constant velocity in a highway (Constant Turn Rate Velocity, CTRV) where only a single path is allowed.

Recently, **Learning-Based** MP [3, 6, 12, 20, 28–31] have become increasingly popular since they are able not only to take into account these above-mentioned factors but also consider interaction-related factors (like agent-agent [32], agent-map [12] and map-map [15]) in such a way the algorithm can adapt to more complex traffic scenarios (intersections, sudden breaks and accelerations, etc). SoPhie [2] is one of the early works on social interactions modelling and forecasting. Methods based on Graph Neural Networks (GNNs) [14, 15, 21] have shown very promising results. MultiPath [13] uses ConvNets as encoder and adopts predefined trajectory anchors to regress multiple possible future trajectories. HOME [5, 14] presented a novel representation for multi-modal trajectory prediction, where the model takes as input the context (HD map) and history of past trajectories, and generates a 2D heatmap of the agent's possible future trajectories. GoalNet [33] identifies possible goal points *i.e.* potential per-agent endpoints, and uses a prediction head for each. In this direction, TNT [31] first samples potential goal points along the lane and generates trajectories conditioned on the high-scored goals. DenseTNT [34] also proposes a trajectory prediction model to output a set of trajectories from a set of inferred goal candidates. However, even DL methods struggle to model accurately the multi-modal nature of the trajectories [10].

A fundamental piece in MP is the **HD map**, methods usually process the map using rasterization and obtaining a multi-channel image with the information encoded though the different channels [5]. Usually this processing is done using a CNN as feature extractor. IntentNet [12] uses a CNN-based detector to extract features from rasterized maps. MultiPath [13] uses a "Scene CNN" to extract mid-level features. Nevertheless, these CNN-based methods suffer from expensive computation when extracting features of graph-structured maps. Moreover, they are limited by the receptive field of the CNN *e.g.* local kernels struggle to process long lanes. VectorNet [22] is one of the first works in this direction, the authors propose to encode map elements and actor trajectories as polylines and then uses a global interactive graph to fuse the actors motion histories, maps, and interactions. We find especially related LaneGCN [15], a method that constructs a map node graph and proposes a novel graph convolution. On top of that, we explore to include physics-based heuristic methods to add a preliminary plausible future area in an interpretable way.

In terms of actors interactions, we make use of **Graph-based** methods [21], which construct graph-structured representations from the HD maps, preserving the connectivity of lanes, and therefore the geometry of the scene.

We use as baseline method Wang*et al.* GANet [24]. This work combines LaneGCN [15] map processing with a goal area prediction to ensure the predicted multi-modal trajectories are plausible.

Datasets Large-scale annotated datasets have been proposed to impulse the research on the MP task. The Waymo Open Motion Prediction [35] and NuScenes Prediction [36] datasets offer thousands of real driving scenarios and exhaustive annotations.

In this work, we focus on the Argoverse 2 [1] Motion Forecasting dataset and benchmark [1], which improves the previous one by spanning +2000km over six different cities. The Argoverse 2 dataset is a high-quality motion forecasting dataset, the real driving scenario are paired with the corresponding local HD map. Compared to Argoverse 1, the scenarios in Argoverse 2 are approximately twice longer and more diverse.

3. Method

Considering the trade-off between curated input data and complexity, we aim to achieve competitive results on the Argoverse 2 Motion Forecasting Benchmark [1] using: (i) the agents past trajectories, additional metadata and their corresponding interactions, (ii) HD Map graph information and (iii) a simplified representation of the agents feasible area as an extra input, which we mention as heuristic proposals. In both map inputs (HD Map graph and heuristic proposals) we include topological, geometric and semantic information to compute the physical context. Fig. 2 shows an overview of our final approach.

Formulation. Given a sequence of past trajectories $a_P = [a_{-T'+1}, a_{-T'+2}, ..., a_0]$ for an agent, we aim to predict its future steps $a_F = [a_1, a_2, ..., a_T]$ up to a fixed time step T. Running in a specific traffic scenario, each actor will interact with static HD maps m and the other dynamic actors. Therefore, the probabilistic distribution that we want to capture is $p(a_F|m, a_P, a_P^O)$, where a_P^O denotes the other actors' observed states. The output of our model is $A_F = \{a_F^k\}_{k\in[0,K-1]} = \{(a_1^k, a_2^k, ..., a_T^k)\}_{k\in[0,K-1]}$ for each actor, while motion forecasting tasks and subsequent decision modules usually expect us to output a set of trajectories. TNT [31]-like methods' distribution can be approximated as

$$\sum_{\tau\in T(m,a_P,a_P^O)} p(\tau|m, a_P, a_P^O)p(a_F|\tau, m, a_P, a_P^O) \quad (1)$$

where $T(m, a_P, a_P^O)$ is the space of candidate goals depending on the driving context. However, the map space m is large, and the goal space $T(m, a_P, a_P^O)$ requires careful design. In that sense, some methods expect to accurately predict the actor motion by extracting good features. For example, LaneGCN [15] tries to approximate $p(a_F|m, a_P, a_P^O)$ by modeling $p(a_F|M_{a_0}, a_P, a_P^O)$, where M_{a_0} is a "local" map features that is related to the actor state a_0 at final observed step $t = 0$. To extract M_{a_0}, they use a_0 as an anchor to retrieve its surrounding map elements and aggregate their features. As stated by [24], computing the local map information is only a part of the solution, but also proposing preliminary guidance for the model in a heuristic way, as well as calculating the goal area maps information using DL, may be of great importance for accuracy trajectory prediction. Then, our future probability distribution is enhanced by these preliminary preprocessed proposals and predicted goals as anchors to explicitly aggregate their surrounding map features as goal areas.

3.1. Preprocessing

As proposed by multiple methods [15] [37], we consider only the vehicles that are observable at *t=0*, handling those agents that are not observed over the full sequence spectrum (observation length = obs_{len} + prediction length = $pred_{len}$) by concatenating a binary flag b_i^t that indicates if the agent is padded or not. In particular, we filter the static elements and track fragments scored by Argoverse 2 to get only the most relevant traffic agents, reducing the number of agents to be considered in complex traffic scenarios. To further leverage the symmetries of the problem, we employ a scene representation that is agnostic about the translation of the global coordinate frame as well as rotation invariant since we translate and rotate both the map and social coordinates

[1]https://eval.ai/web/challenges/challenge-page/2006/overview

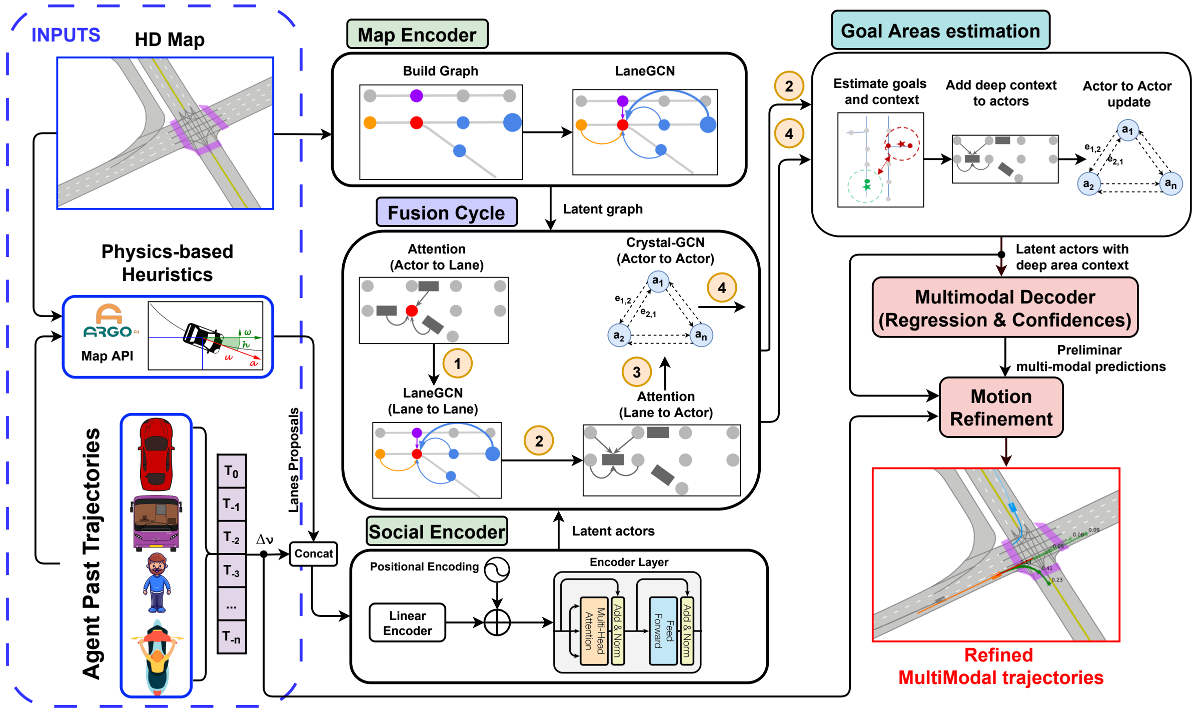

Figure 2. Overview of our Motion Prediction pipeline. We distinguish: 1) **Social Encoder**, which uses the agent past trajectories (relative displacements and additional metadata such as the type (*e.g.* car, cyclist, pedestrian) and category, from less to more important) and the corresponding heuristic lane proposals to compute the social features, 2) **Map Encoder**, that constructs a lane graph from the HD Map and uses a LaneConv operator [15] to extract lane node features, 3) **Fusion Cycle**, responsible for fusing agents and map latent features, 4) **Goal Areas estimation**, to predict some goals and their surrounding (area) features are aggregated to the agents, 5) **Multimodal Decoder**, which uses the latent actors with deep area context to generate reliable multi-modal predictions and 6) **Motion Refinement**, in charge of enhancing the quality of the future trajectories taking into account the past trajectories, actors latent features and preliminary predicted trajectories

considering the origin and orientation to align with x-axis are the features of the focal track at timestep obs_{len}. Moreover, instead of using absolute 2D-BEV (*xy* plane), the input for the agent *i* is a series of relative displacements:

$$\Delta \boldsymbol{\nu}_i^t = \boldsymbol{\nu}_i^t - \boldsymbol{\nu}_i^{t-1} \tag{2}$$

Where $\boldsymbol{\nu}_i^t$ represents the state vector (in this case, *xy* position of the agent *i* at timestamp *t*). We additionally compute and codify the object type (bus, pedestrian, vehicle, cyclist) and track category (unscored, scored and focal track) as additional metadata.

In terms of HD Map graph preprocessing, we follow the same principles than other well-established baselines [15] [24] by adopting simple form of vectorized map data as our rep- resentation of HD maps. In this case, the map data is represented as a set of polylines (lanes) and their connectivity, where each lane contains a centerline (sequence of 2D Bird's Eye View points), arranged following the lane direction. For any two lanes which are directly reachable, 4 types of connections are given: predecessor, successor, left neighbour and right neighbour.

Heuristic Proposals In order to compute the heuristic proposals for each agent, first of all we filter the trajectory using Least Squares (2nd order) and Savitzky-Golais filters to obtain smooth values of the velocity, acceleration and orientation of the corresponding agent in the last observation frame, calculating the future travelled distance by extending the smoothed 2nd-order polynom T seconds. Then, we get all lane candidates within a bubble, given the agent last observation and Manhattan distance, expanding the bubble until at least 1 lane is found. Once we have some preliminary proposals, we employ the Depth First Search (DFS) algorithm to get all successor and predecessor candidates, merging the past and future candidates and removing the overlapping ones.

After removing the overlapping lanes, we select the top-M lanes based on the scores as proposed by [38], illustrating a preliminary interpretable motion prediction area in front of the agent. We include additional metadata such as lane type (bus, bike, vehicle), presence of intersection or boundaries topology (dash, solid, yellow), along with the centerline relative displacements. Finally, these heuristic proposals are interpolated using a 1st order spline from last agent observation as start point to an endpoint located in the centerline assuming the initially-computed travelled distance in the prediction horizon.

3.2. Encoder and Fusion Cycle

The first stage of MP is driving context encoding, which extracts actors motion features and maps features. In terms of physical context, we adopt LaneGCN [15] backbone to encode the scene context for its outstanding performance. It learns good lane representations which are computationally efficient and preserve map topology. We use a multi-scale LaneConv network to encode the vectorized map data, which is consisted of lane centerlines and their connectivity. We construct a lane node graph from the map data. A lane node is a short lane segment between two consecutive points of the lane centerline, represented by the location (the averaged coordinates of its two endpoints) and the shape (the vector between its two endpoints).

In terms of agent information, several methods [37] [24] [39] [40] make use of Recurrent Neural Networks (such as GRUs and LSTMs). LSTMs became popular because they could solve the problem of vanishing gradients. Nevertheless, they require a lot of resources to get trained and become ready for real-world applications. In particular, they need high memory-bandwidth because of linear layers present in each cell which the system usually fails to provide for. In that sense, we replace LSTM social encoder for an encoder transformer, which is faster than RNN-based models as all the input is ingested once, decreasing the computational complexity. We concatenate the agents past trajectories, additional social metadata and heuristical proposals (including semantic and topologic lanes metadata), which is processed by a linear embedding. Then, positional encoding is added to the output embedding explicitly to retain the information regarding the order of past trajectories and future preliminary steps. Finally, these latent features feed the encoder transformer, leveraging self-attention mechanism and positional embedding to learn complex and dynamic patterns from long-term time series data.

Once both the map and social latent features are computed, we obtain a 2D feature matrix X where each row X_i indicates the feature of the i-th actor, and a 2D matrix Y where each row Y_i indicates the feature of the i-th lane node. Then, in order to combine the latent social and physical information, we make use of the well-established actor-map fusion cycle [15] that transfers and aggregates feature among actors and lane nodes. Nevertheless, at the end of the fusion cycle, instead of using an Actor to Actor (A2A) attention [15] [24] method, which uses standard self-attention to model final actor features, we employs the use of a Graph Convolution Operator (GCN), inspired in the architecture proposed by [37], to further enhance global agent interaction.

3.3. Global Interaction Module

After aggregating the map features to the corresponding actors (Third step in the fusion cycle, as observed in Fig. 2), we compute the interaction among actors in order perform global message-passing by constructing an interaction graph using Crystal-GCN [41] [37], originally developed for the prediction of material properties, allowing to efficiently leverage edge features.

Before creating the **interaction mechanism**, we split the temporal information in the corresponding scenes, taking into account that each traffic scenario may have a different number of agents. The interaction mechanism is defined in [37] as a bidirectional fully-connected graph, where the initial node features $\mathbf{v}_i^{(0)}$ are represented by the latent temporal information for each vehicle $\mathbf{h}_{i,out}$ computed by the motion history encoder. On the other hand, the edges from node *k* to node l is represented as the vector distance ($\mathbf{e}_{k,l}$) between the corresponding agents at t = obs_{len} in absolute coordinates, where the origin of the sequence ($x = 0, y = 0$) is represented by the position of the target at t = obs_{len}:

$$\mathbf{e}_{k,l} = \boldsymbol{\nu}_k^{obs_{len}} - \boldsymbol{\nu}_l^{obs_{len}}, \tag{3}$$

Given the interaction graph (nodes and edges), the Crystal-GCN, proposed by [41], is defined as:

$$\mathbf{v}_i^{(g+1)} = \mathbf{v}_i^{(g)} + \sum_{j=0:j\neq i}^{N} \sigma\left(\mathbf{z}_{i,j}^{(g)}\mathbf{W}_{\mathrm{f}}^{(g)} + \mathbf{b}_{\mathrm{f}}^{(g)}\right) \odot \mu\left(\mathbf{z}_{i,j}^{(g)}\mathbf{W}_{\mathrm{s}}^{(g)} + \mathbf{b}_{\mathrm{s}}^{(g)}\right). \tag{4}$$

This operator, in contrast to many other graph convolution operators [21] [15], allows the incorporation of edge features in order to update the node features based on the distance among vehicles (the closer a vehicle is, the more is going to affect to a particular node). As stated by [37], we use $L_g = 2$ layers of the GNN ($g \in 0, \ldots, L_g$ denotes the corresponding Crystal-GCN layer) with ReLU and batch normalization as non-linearities between the layers. σ and μ are the sigmoid and softplus activation functions respectively. Moreover, $\mathbf{z}_{i,j}^{(g)} = (\mathbf{v}_i^{(g)} || \mathbf{v}_j^{(g)} || \mathbf{e}_{i,j})$ corresponds to the concatenation of two node features in the g_{th} GNN layer and the corresponding edge feature (distance between agents), N represents the total number of agents in the scene and $\mathbf{W}$ and $\mathbf{b}$ the weights and bias of the corresponding layers respectively.

After the interaction graph, each updated node feature $\mathbf{v}_i^{(L_g)}$ contains information about the latent space of other agents, specially those which are closer to that particular node. We can appreciate that instead of applying the previously mentioned self-attention mechanism, where the latent space of an agent pays attention to itself, now we are able to enhance global interaction to model complex behaviours, specially in roundabouts and intersections.

3.4. Decoding Module

3.4.1 Goals Prediction and Filtering

Goals predictions Since we use GANet [24] as baseline method, we follow the same approach to predict goal points *i.e.* potential destination points for each agent in the scene.

We train the goal predictor as [24] -following the same network architecture- using a combination of classification loss and regression loss. Given E predicted goals, we find a positive goal $\hat{e}$ that has the minimum $\mathcal{L}_2$ distance *w.r.t* the ground truth trajectory endpoint — making the predicted goal close to the actual goal as much as possible.

This goal prediction module serves to locate goal areas (*e.g.* plausible areas of movement within the lanes). In practice, a drivers behaviour is highly multi-modal and stochastic *e.g.* the driver can stop, go ahead, turn left or right when approaching an intersection, or accelerate. Therefore, we try to make a multiple-goals prediction.

GoICrop Following Wang *et al.* GANet [24], we choose the predicted goal with the highest confidence among E goals as an anchor. This anchor, by definition, approximates the destination of the agent with the highest probability based on its past trajectories and the scene context.

Hence, we use the predicted anchors and apply a GoICrop [24] filtering to implicitly model actors' interactions in the future. This ROI (Regions of Interest) filter allows to select the goal points (anchors) for each agent, considering the their probable interations.

3.4.2 Trajectory Prediction

Finally, we take the updated actor features X as input to predict K final future trajectories and their confidence scores. Following our baseline [24] we use a standard multi-modal decoder with one regression branch estimating the trajectories, and one classification branch predicting the corresponding confidences.

3.5. Motion refinement

We finally apply motion refinement as proposed by Liu *et al.* [39] to improve further the temporal consistency and output more accurate future trajectories. The goal is to reduce the offset between ground truth trajectory Y and predicted trajectory $\hat{Y}$. We define this offset as $\Delta Y = Y - \hat{Y}$.

Using this approach, an MLP model is trained to minimize the offset by predicting a residual R that is added to the original trajectory *i.e.* we use L_2 loss to optimize the offset as follows:

$$\mathcal{L}_{\text{off}} = ||Y - \hat{Y} - \hat{R}||_2 = ||\Delta Y - \hat{R}||_2. \tag{5}$$

Note that this method can be applied to the pre-trained model from previous stages, which is completely functional, as the main function is to improve the output trajectories.

4. Experiments

Experimental settings We use the Argoverse 2 [1] dataset and benchmark described in Section 2. For each real driving scenario we have the corresponding local HD map, past trajectories of the agents, metadata about the agents (*e.g.* the ype of agent: cyclist, pedestrian, car), and topological information about the scene. Each scenario is 11 seconds long. We consider five seconds of the past trajectory (also known as motion history), and we predict the next six seconds.
Metrics. We follow the widely used evaluation metrics [21, 34, 45]. Specifically, MR is the ratio of predictions where none of the predicted K trajectories is within 2.0 meters of ground truth according to the endpoint displacement error. Minimum Final Displacement Error (minFDE) is the L2 distance between the endpoint of the best-forecasted trajectory and the ground truth. Minimum Average Displacement Error (minADE) is the average L2 distance between the best-forecasted trajectory and the ground truth.
Implementation. We train our model on 2 A100 GPUs using a batch size of 128 with the Adam optimizer for 42 epochs. The initial learning rate is 1 x 10-3, decaying to 1 x 10-4 at 32 epochs. The latent dimension (regarding map and social features) in most of our experiments is 128. The number of attention heads in the social encoder and motion refinement is 8. The training setup including loss functions follows GANet [24] official implementation as our baseline. We set the number of lane proposals to 3, since this is typically the number of maximum manoeuvres an agent can carry out (straight, left, right). If the heuristic method computes less lanes than M=3, we pad with zeros.
Augmentations (i) Dropout and swapping random points from the past trajectory, (ii) point location perturbations under a $\mathcal{N}(0, 0.2)$ [m] noise distribution [45].

4.1. Results and Ablation studies

Tables 1 and 2 present the results obtained on the Argoverse 2 Motion Forecasting validation and test sets, respectively. We achieve near state-of-the-art performance in both

Table 1. Comparison of methods in the **Argoverse 2 Validation** Set. We show the number of parameters for each model, prediction metrics (minADE, minFDE and brier-minFDE) for the multimodal scenario (k=6) and runtime. Runtime was measured on a single GPU A100-SXM4 (using batch 128). Our experiments are indicated using †. We use as baseline method GANet [24].

Method	Map	# Par. (M)	minADE (m) ↓	minFDE (m) ↓	brier-minFDE (m) ↓	Runtime (ms) ↓
GANet [24]	Yes	6.2	**0.806**	**1.402**	**2.02**	1612
GANet w/o Map Decoder [24]	Yes	5.7	0.84	1.55	2.18	1353
GANet w/o Goal Areas [24]	Yes	4.5	0.87	1.66	2.29	1134
GANet w/o map [24]	No	1.79	1.034	2.212	2.825	838
† CRAT-Pred [37]	No	**0.53**	1.31	2.78	3.65	**223**
† Ours-base: GANet [24] ActorNet → Attention Transformer	Yes	5.0	0.83	1.45	2.07	923
† Ours-m: Ours-base + A2A → C-GCN + Metadata	Yes	4.74	0.82	1.43	2.05	892
† Ours-s: Ours-base + A2A → C-GCN + Metadata (64 latent size)	Yes	1.2	0.88	1.53	2.15	893
† Ours: Ours-m + Proposals + Motion Refinement	Yes	4.92	0.81	1.42	2.04	946

Table 2. Results on **Argoverse 2 Test** dataset. The "-" denotes that this result was not reported in their paper. Some numbers are borrowed from [24]. For all the metrics, the lower, the better.

Method	b-minFDE (K=6)	MR (K=6)	minFDE (K=6)	minADE (K=6)	minFDE (K=1)	minADE (K=1)	MR (K=1)
DirEC	3.29	0.52	2.83	1.26	6.82	2.67	0.73
drivingfree	3.03	0.49	2.58	1.17	6.26	2.47	0.72
LGU	2.77	0.37	2.15	1.05	6.91	2.77	0.73
Autowise.AI(GNA)	2.45	0.29	1.82	0.91	6.27	2.47	0.71
Timeformer [42]	2.16	0.20	1.51	0.88	4.71	1.95	0.64
QCNet	2.14	0.24	1.58	0.76	4.79	1.89	0.63
OPPred w/o Ensemble [43]	2.03	0.180	1.389	0.733	4.70	1.84	0.615
TENET w/o Ensemble [44]	2.01	-	-	-	-	-	-
Polkach(VILaneIter)	2.00	0.19	1.39	**0.71**	4.74	1.82	0.61
GANet	**1.969**	**0.171**	**1.352**	0.728	**4.475**	**1.775**	**0.597**
Ours	1.98	0.185	1.37	0.73	4.53	1.79	0.608

sets, which is on-pair with the most promising pipelines, while using notably less parameters. As stated throughout this work, we focus on applying efficient methods to help understand future interactions among the different agents, reducing the number of parameters and inference time.

We can appreciate in Table 1 the huge influence of the physical context both in terms of accuracy and runtime. GANet [24] shows the best multimodal prediction metrics, with an approximate amount of 6.2M of parameters of 1612 ms given a batch size of 128 traffic scenarios and an average number of 30 agents per scene. As expected, progressively removing the map influence (remove map from decoder, remove goal areas estimation) in the model we decrease the MP performance with a noticeable parameter decrease.

In our case, we study the influence of substituting the modified ActorNet [15, 24] social encoder proposed by GANet, which uses RNNs. Our proposal replaces these by a Linear embedding, a Positional Encoding and Encoder Transformer. Moreover, we add the aforementioned agent metadata (object type and track category), and we substitute the Actor to Actor attention of the fusion cycle for a GCN [37] operator to enhance agents global interaction. It can be appreciated how we obtained a similar performance (both with 128 latent dimension in *Ours-m*, and 64 latent in *Ours-s*), reducing the parameters and inference time.

Finally, our best model, which includes heuristic proposals that serve as a preliminary multi-modal guidance for the model and motion refinement to improve the quality of the final predictions, obtains a performance on pair with [24], reducing the number of parameters and inference time about 21% and 41% respectively. We can appreciate in Table 2 how our model generalizes well in the test set, with results (both in uni-modal and multi-modal prediction) up-to-pair with other state-of-the-art algorithms.

4.2. Qualitative results

We provide advanced qualitative samples in Figure 3, where we show the HD Map of real traffic scenes, heuristic trajectory proposals in the form of centerlines, and the multimodal predictions from our model including their respective confidences (the higher, the most probable).

As we discussed in 3, we designed our model to ensure realistic predictions. We can appreciate that all the predictions are plausible and constrained to the scene geometry *e.g.* lane distribution and centerlines. We believe our heuristic proposals help to regularize the model and pro-

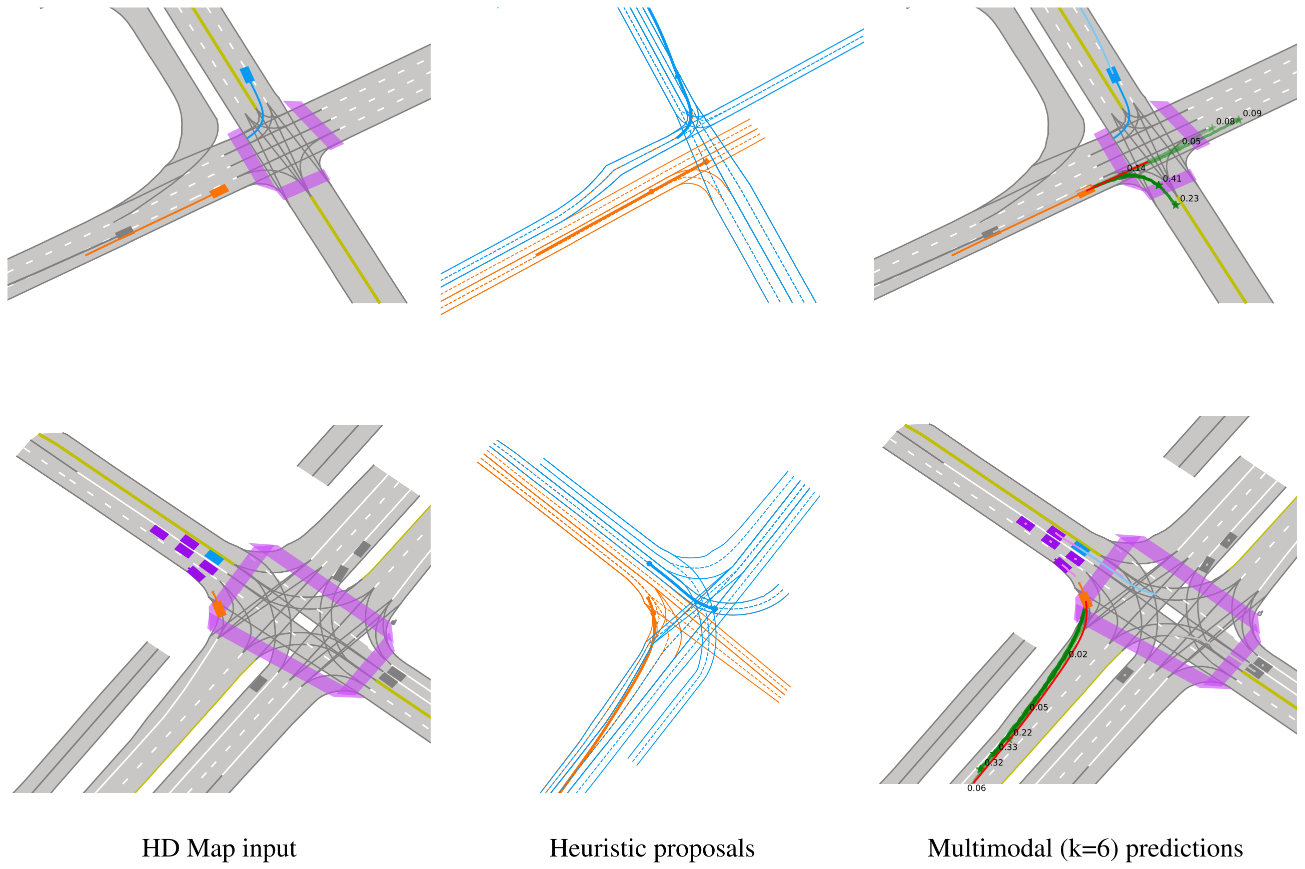

Figure 3. Qualitative Results on challenging scenarios using our best model. We represent: our vehicle (**ego**), the **focal agent**, the **relevant agents** in the scene, and **other agents**. We can also see the **ground-truth** trajectory of the target agent, our **multimodal predictions** (with the corresponding **confidences**). We also highlight the most important topology of the road, such as pedestrian crossing and boundaries mark type. We show, from left to right, a general view of the traffic scenario (including and map information), the heuristic proposals for each agent (we only include the **ego** and **focal agent** for simplicity) and the multimodal prediction (k = 6) for the **focal agent**, including the corresponding confidences (the higher, the most probable)

duce realistic predictions that would ensure traffic safety. For simplicity, we only illustrate the heuristic proposals for the focal agent and ego-vehicle. We believe our software for qualitative analysis of MP models on the well-known Argoverse 2.0 [1] is fundamental and a core contribution.

5. Conclusion

In this work we solve the challenging problem of Multi-Agent Motion Prediction in real driving scenarios. We present an end-to-end pipeline that combines Deep Learning (DL) and heuristic scene understanding. Our model uses as input the map of the scene, the past trajectories of the agents, and additional information about the scene geometry and agents e.g., type of agent, lane distribution. We propose a model that integrates attention mechanisms with GNNs, heuristic goals, and a motion refinement module to further improve temporal consistency. We achieve SOTA results on the Argoverse 2 Motion Forecasting Benchmark reducing in millions of parameters previous methods such as GANet, and improving over LaneGCN. Our code is publicly available. As future works, we plan to include map-adaptive lane-loss to improve diverse multiple motion prediction and explore knowledge-distillation to improve the efficiency for real-world deployment.

Acknowledgements This work has been funded in part from the Spanish MICINN/FEDER through the Artificial Intelligence based modular Architecture Implementation and Validation for Autonomous Driving (AIVATAR) project (PID2021-126623OB-I00), Electric Automated Vehicle for Aging Drivers (AVAD) project (PDC2022-133470-I00) and from the RoboCity2030-DIH-CM project (P2018/NMT- 4331), funded by Programas de actividades I+D (CAM), cofunded by EU Structural Funds and Scholarship for Introduction to Research activity by University of Alcalá.

References

[1] 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," *arXiv preprint arXiv:2301.00493*, 2023. 1, 3, 6, 8

[2] A. Sadeghian, V. Kosaraju, A. Sadeghian, N. Hirose, H. Rezatofighi, and S. Savarese, "Sophie: An attentive gan for predicting paths compliant to social and physical constraints," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1349–1358, 2019. 1, 2

[3] R. Mahjourian, J. Kim, Y. Chai, M. Tan, B. Sapp, and D. Anguelov, "Occupancy flow fields for motion forecasting in autonomous driving," *IEEE Robotics and Automation Letters*, 2022. 1, 2

[4] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data," in *European Conference on Computer Vision*, pp. 683–700, Springer, 2020. 1, 2

[5] 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)*, pp. 500–507, IEEE, 2021. 1, 2, 3

[6] P. Dendorfer, A. Osep, and L. Leal-Taixé, "Goal-gan: Multimodal trajectory prediction based on goal position estimation," in *Proceedings of the Asian Conference on Computer Vision*, 2020. 1, 2

[7] C. Katrakazas, M. Quddus, W.-H. Chen, and L. Deka, "Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions," *Transportation Research Part C: Emerging Technologies*, vol. 60, pp. 416–442, 2015. 1, 2

[8] C. Gómez-Huélamo, L. M. Bergasa, R. Gutiérrez, J. F. Arango, and A. Díaz, "Smartmot: Exploiting the fusion of hdmaps and multi-object tracking for real-time scene understanding in intelligent vehicles applications," in *2021 IEEE Intelligent Vehicles Symposium (IV)*, pp. 710–715, IEEE, 2021. 1, 2

[9] J. Hong, B. Sapp, and J. Philbin, "Rules of the road: Predicting driving behavior with a convolutional model of semantic interactions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8454–8462, 2019. 1

[10] Y. Huang, J. Du, Z. Yang, Z. Zhou, L. Zhang, and H. Chen, "A survey on trajectory-prediction methods for autonomous driving," *IEEE Transactions on Intelligent Vehicles*, 2022. 2

[11] N. Deo and M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1468–1476, 2018. 2

[12] S. Casas, W. Luo, and R. Urtasun, "Intentnet: Learning to predict intention from raw sensor data," in *Conference on Robot Learning*, pp. 947–956, PMLR, 2018. 2, 3

[13] Y. Chai, B. Sapp, M. Bansal, and D. Anguelov, "Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction," in *Proceedings of the Conference on Robot Learning* (L. P. Kaelbling, D. Kragic, and K. Sugiura, eds.), vol. 100 of *Proceedings of Machine Learning Research*, pp. 86–99, PMLR, 30 Oct–01 Nov 2020. 2, 3

[14] T. Gilles, S. Sabatini, D. Tsishkou, B. Stanciulescu, and F. Moutarde, "Gohome: Graph-oriented heatmap output for future motion estimation," *arXiv preprint arXiv:2109.01827*, 2021. 2

[15] M. Liang, B. Yang, R. Hu, Y. Chen, R. Liao, S. Feng, and R. Urtasun, "Learning lane graph representations for motion forecasting," in *European Conference on Computer Vision*, pp. 541–556, Springer, 2020. 2, 3, 4, 5, 7

[16] Z. Huang, X. Mo, and C. Lv, "Multi-modal motion prediction with transformer-based neural network for autonomous driving," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 2605–2611, IEEE, 2022. 2

[17] Y. Liu, J. Zhang, L. Fang, Q. Jiang, and B. Zhou, "Multimodal motion prediction with stacked transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7577–7586, 2021. 2

[18] 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. 2

[19] Y. B. Can, A. Liniger, O. Unal, D. Paudel, and L. Van Gool, "Understanding bird's-eye view of road semantics using an onboard camera," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3302–3309, 2022. 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," *arXiv preprint arXiv:2111.14973*, 2021. 2

[21] 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)*, pp. 532–539, IEEE, 2021. 2, 3, 5, 6

[22] 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*, pp. 11525–11533, 2020. 2, 3

[23] R. Walters, J. Li, and R. Yu, "Trajectory prediction using equivariant continuous convolution," *arXiv preprint arXiv:2010.11344*, 2020. 2

[24] M. Wang, X. Zhu, C. Yu, W. Li, Y. Ma, R. Jin, X. Ren, D. Ren, M. Wang, and W. Yang, "Ganet: Goal area network for motion forecasting," *arXiv preprint arXiv:2209.09723*, 2022. 2, 3, 4, 5, 6, 7

[25] J. Bock, R. Krajewski, T. Moers, S. Runde, L. Vater, and L. Eckstein, "The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections," in *2020 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1929–1934, IEEE, 2019. 2

[26] Z. Tang, M. Naphade, M.-Y. Liu, X. Yang, S. Birchfield, S. Wang, R. Kumar, D. Anastasiu, and J.-N. Hwang, "Cityflow: A city-scale benchmark for multi-target multi-camera vehicle tracking and re-identification," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, p. 8797–8806, June 2019. 2

[27] M. Naphade, S. Wang, D. C. Anastasiu, Z. Tang, M. Chang, Y. Yao, L. Zheng, M. S. Rahman, A. Venkatachalapathy, A. Sharma, Q. Feng, V. Ablavsky, S. Sclaroff, P. Chakraborty, A. Li, S. Li, and R. Chellappa, "The 6th ai city challenge," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 3346–3355, IEEE Computer Society, June 2022. 2

[28] N. Deo and M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1468–1476, 2018. 2

[29] N. Rhinehart, R. McAllister, K. Kitani, and S. Levine, "Precog: Prediction conditioned on goals in visual multi-agent settings," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2821–2830, 2019. 2

[30] B. Ivanovic, K.-H. Lee, P. Tokmakov, B. Wulfe, R. McAllister, A. Gaidon, and M. Pavone, "Heterogeneous-agent trajectory forecasting incorporating class uncertainty," *arXiv preprint arXiv:2104.12446*, 2021. 2

[31] 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, 3

[32] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social gan: Socially acceptable trajectories with generative adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2255–2264, 2018. 2

[33] L. Zhang, P.-H. Su, J. Hoang, G. C. Haynes, and M. Marchetti-Bowick, "Map-adaptive goal-based trajectory prediction," *arXiv preprint arXiv:2009.04450*, 2020. 2

[34] J. Gu, Q. Sun, and H. Zhao, "Densetnt: Waymo open dataset motion prediction challenge 1st place solution," *arXiv preprint arXiv:2106.14160*, 2021. 2, 6

[35] S. Ettinger, S. Cheng, B. Caine, C. Liu, H. Zhao, S. Pradhan, Y. Chai, B. Sapp, C. R. Qi, Y. Zhou, *et al.*, "Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9710–9719, 2021. 3

[36] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11621–11631, 2020. 3

[37] J. Schmidt, J. Jordan, F. Gritschneder, and K. Dietmayer, "Crat-pred: Vehicle trajectory prediction with crystal graph convolutional neural networks and multi-head self-attention," *arXiv preprint arXiv:2202.04488*, 2022. 3, 5, 7

[38] S. Khandelwal, W. Qi, J. Singh, A. Hartnett, and D. Ramanan, "What-if motion prediction for autonomous driving," *arXiv preprint arXiv:2008.10587*, 2020. 4

[39] M. Liu, H. Cheng, L. Chen, H. Broszio, J. Li, R. Zhao, M. Sester, and M. Y. Yang, "Laformer: Trajectory prediction for autonomous driving with lane-aware scene constraints," *arXiv preprint arXiv:2302.13933*, 2023. 5, 6

[40] C. Gómez-Huélamo, M. V. Conde, and M. Ortiz, "Exploring map-based features for efficient attention-based vehicle motion prediction," *arXiv preprint arXiv:2205.13071*, 2022. 5

[41] T. Xie and J. C. Grossman, "Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties," *Physical review letters*, vol. 120, no. 14, p. 145301, 2018. 5

[42] 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. 7

[43] C. Zhang, H. Sun, C. Chen, and Y. Guo, "Banet: Motion forecasting with boundary aware network," *arXiv preprint arXiv:2206.07934*, vol. 2, 2022. 7

[44] 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. 7

[45] M. Ye, T. Cao, and Q. Chen, "Tpcn: Temporal point cloud networks for motion forecasting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11318–11327, 2021. 6