ProphNet: Efficient Agent-Centric Motion Forecasting with Anchor-Informed Proposals

Xishun Wang Tong Su Fang Da Xiaodong Yang*
QCraft

Abstract

Motion forecasting is a key module in an autonomous driving system. Due to the heterogeneous nature of multi-sourced input, multimodality in agent behavior, and low latency required by onboard deployment, this task is notoriously challenging. To cope with these difficulties, this paper proposes a novel agent-centric model with anchor-informed proposals for efficient multimodal motion prediction. We design a modality-agnostic strategy to concisely encode the complex input in a unified manner. We generate diverse proposals, fused with anchors bearing goal-oriented scene context, to induce multimodal prediction that covers a wide range of future trajectories. Our network architecture is highly uniform and succinct, leading to an efficient model amenable for real-world driving deployment. Experiments reveal that our agent-centric network compares favorably with the state-of-the-art methods in prediction accuracy, while achieving scene-centric level inference latency.

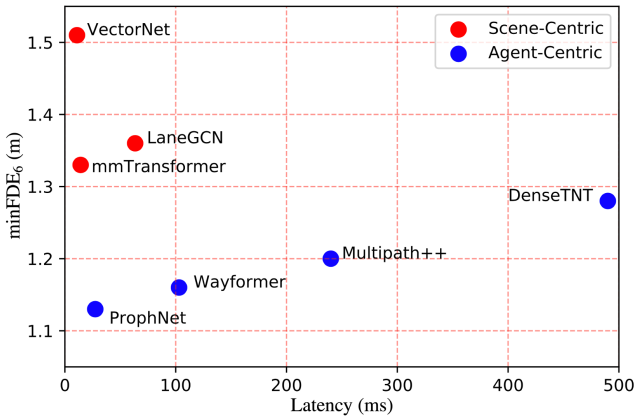

Figure 1. Overview of accuracy-and–latency trade-off for the task of motion forecasting on Argoverse-1. ProphNet outperforms the state-of-the-art methods in prediction accuracy and considerably speeds up the agent-centric inference latency, leading to the best balance between accuracy and latency.

1. Introduction

Predicting the future behaviors of various road participants is an essential task for autonomous driving systems to be able to safely and comfortably operate in dynamic driving environments. However, motion forecasting is extremely challenging in the sense that (i) the input consists of interrelated modalities from multiple sources; (ii) the output is inherently stochastic and multimodal; and (iii) the whole prediction pipeline must fulfill tight run time requirements with a limited computation budget.

A motion forecasting model typically collects the comprehensive information from perception signals and high-definition (HD) maps, such as traffic light states, motion history of agents, and the road graph [11, 13, 14, 23]. Such a collection of information is a heterogeneous mix of static and dynamic, as well as discrete and continuous elements. Moreover, there exist complex semantic relations between these components, including agent-to-agent, agent-to-road, and road-to-road interactions. Previous methods in the field usually model this diverse set of input via an equally intricate system with modality-specific designs. LaneGCN [9] adopts four sub-networks to separately process the interactions of agent-to-agent, agent-to-road, road-to-agent and road-to-road. MultiPath++ is developed in [21] to employ multi-context gating to first capture the interactions between a target agent and other agents, and then fuse them with the map in a hierarchical manner.

Due to the unknown intent of an agent, motion prediction output is highly multimodal by nature. It is often impossible to assert with full certainty whether a vehicle will go straight or turn right as it approaches an intersection. The motion prediction model is therefore required to accurately model the underlying probability distribution of future behaviors. Recently, there have been various efforts on improving output multimodality. TNT [30] introduces three stages including goal (endpoint of a predicted trajectory) prediction, goal-conditioned motion estimation, and trajectory scoring. DenseTNT is proposed in [7] to improve upon the former by predicting from dense goal candidates

*Corresponding author xiaodong@qcraft.ai

and conducting an iterative offline optimization algorithm to generate multi-future pseudo labels. A region based training strategy is developed in mmTransformer [12] to manually partition a scene into subregions to enforce predictions to fall into different regions to promote multimodality.

In a practical autonomous driving system, the whole prediction pipeline typically operates at a frequency of 10Hz, including (i) feature extraction from upstream modules, (ii) network inference, and (iii) post-processing. Therefore, a motion forecasting network is required to meet the strict inference latency constraint to be useful in the real-world setting. While the recent agent-centric paradigm [15, 21] is trending and has made remarkable progress on improving accuracy, its latency is dramatically increased compared to its scene-centric counterpart [5, 12], as shown in Figure 1. This is because agent-centric models compute scene representations with respect to each agent separately, meaning the amount of features to process is dozens of times larger than that with scene-centric models. This substantially increases the latency of the whole prediction pipeline and is computationally prohibitive for onboard deployment.

In light of these observations, we present **ProphNet**: an efficient agent-centric motion forecasting model that fuses heterogeneous input in a unified manner and enhances multimodal output via a design of anchor-informed proposals. In contrast to the existing complex modality-specific processing [9, 21], we develop a modality-agnostic architecture that combines the features of agent history, agent relation and road graph as the agent-centric scene representation (AcSR), which directly feeds to a unified self-attention encoder [22]. In this way, we motivate the self-attention network to learn the complex interactions with minimum inductive bias. Additionally, we propose the anchor-informed proposals (AiP) in an end-to-end learning fashion to induce output multimodality. In our approach, *proposals* are the future trajectory embeddings conjectured exclusively from agent history, and *anchors* refer to the goal embeddings learned from AcSR. In such a manner, the network learns to first generate proposals to maximize diversity without environmental restrictions, and then select and refine after absorbing anchors that carry rich goal-oriented contextual information. Based on random combinations of AiP, we further introduce the hydra prediction heads to encourage the network to learn complementary information and meanwhile perform ensembling readily.

As illustrated in Figure 2, this design formulates a succinct network with high efficiency in terms of both architecture and inference. Instead of employing a self-attention encoder for each input modality [9, 12], the self-attention on AcSR unifies the learning of intra- and inter-modality interactions in a single compact space, which allows the network to assign associations within and across input modalities with maximum flexibility. Our model also considerably reduces inference latency and performs prediction with a peak latency as low as 28ms. In addition, rather than heuristically predefining or selecting goals to strengthen output multimodality [1, 7, 30], AiP is learned end-to-end to infuse goal based anchors to proposals that are deliberately produced with diverse multimodality. In the end, together with the hydra prediction heads, our network is capable of generating future trajectories with rich variety.

Our main contributions are summarized as follows. First, we develop an input-source-agnostic strategy based on AcSR to model heterogeneous input and simplify network architecture, making the model amenable for real-world driving deployment. Second, we propose a novel framework that couples proposal and anchor learning end-to-end through AiP to promote output multimodality. Third, we introduce hydra prediction heads to learn complementary prediction and explore ensembling. Fourth, we show that our network, as an agent-centric model, achieves state-of-the-art accuracy on the popular benchmarks while maintaining scene-centric level low inference latency.

2. Related Work

Representation Paradigms. Motion forecasting methods can be roughly categorized into two types according to the spatial feature representation: rasterized [1, 18, 20] and vectorized [5, 7, 27]. Rasterized methods render agent states and HD maps as color-coded attributes, and use image-oriented networks (e.g., CNN) to encode scene context. Vectorized methods represent agent dynamics and map elements in their vectorized form. Combined with the attention mechanism, vectorized approaches have dominated the leading results recently. On the other hand, regarding the reference points for feature encoding, methods fall into two groups: scene-centric [5, 16] and agent-centric [15, 30]. In the former, scene representation is extracted once for all agents in a shared coordinate frame, while the latter repeatedly builds a normalized coordinate frame for each agent and extracts corresponding scene features. As a result, the scene-centric models are generally more computationally efficient, and the agent-centric models deliver better accuracy.

Modeling Heterogeneous Input. Modeling the complex interactions among diverse types of input is vital for effectively capturing the scene context. LaneGCN [9] uses graph convolution to improve road graph representation, and proposes four input-specific fusion modules for interaction modeling. Based on LaneGCN, BANet [29] further incorporates more road graph elements and interaction fusion blocks, while PAGA [4] augments map encoding with attention between non-adjacent map elements. SceneTransformer [16] develops multi-axis attention to model spatial and temporal information, where agent-to-agent and agent-to-road interactions are learned in separate modules. Wayformer [15] also adopts multi-axis attention and studies fus-

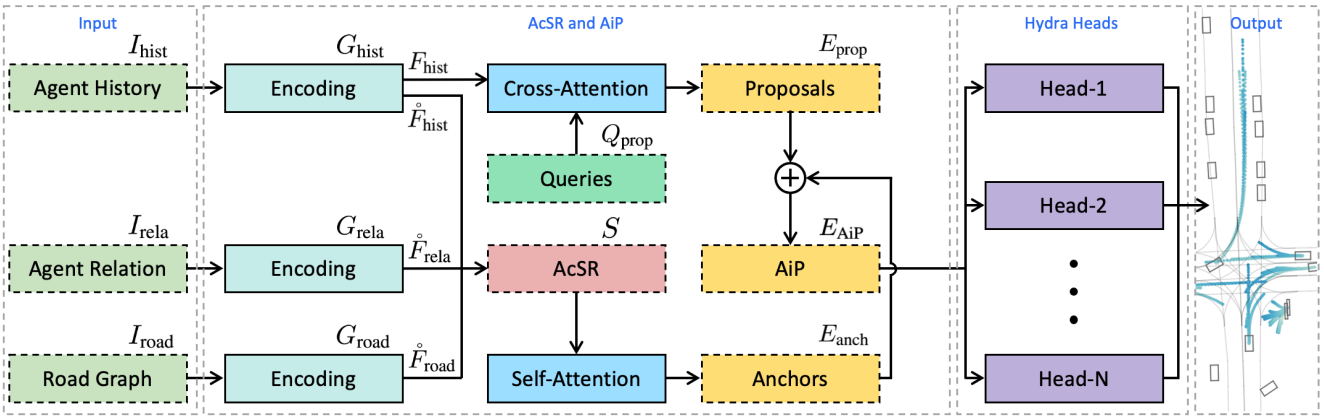

Figure 2. A schematic overview of ProphNet. We uniformly encode the heterogeneous input by gMLP, and combine the compact encoding features to form a unified representation space AcSR. We generate proposals through cross-attention between learnable queries and full history encoding. Anchors are learned based on self-attention of AcSR. We introduce AiP by integrating proposals and anchors, and randomly select subsets from AiP to feed into hydra prediction heads to output the final prediction. We use the solid and dashed boxes to indicate operators and operands in the network, respectively.

ing multi-sourced input at different stages. Most previous works focus on explicitly processing and combining individual input sources to better model their interactions, thus inevitably resulting in increased model complexity.

Enhancing Multimodal Output. Extensive research efforts have been made in multimodal prediction. Planning based methods [19] apply basic planning rules to generate plenty of feasible trajectories following the road graph and kinematic constraints as trajectory candidates. In [12] mmTransformer uses a region based training scheme placing predictions in manually defined regions to enrich the output. MultiPath [1] clusters trajectories from training data as prediction anchors to prevent mode collapse. MultiPath++ [21] learns a fixed set of hidden anchor embeddings to capture different modalities. TNT [30] and DenseTNT [7] rely on sampling sparse and dense goal candidates to achieve good coverage. We aim to induce multimodality through the proposed anchor-informed proposals that can be learned fully end-to-end and is capable of adapting to specific input.

3. Method

Figure 2 illustrates the overall architecture of ProphNet, which involves encoding heterogeneous input into an agent-centric scene representation, end-to-end generation of proposals and anchors, coupling them to form anchor-informed proposals, as well as hydra prediction heads to produce multimodal and complementary trajectories.

3.1. Input Representation

ProphNet performs motion forecasting based on multi-sourced input, including the motion history of target agent, other agents in the surroundings, and road graph. Representations of different input are detailed as follows.

Agent History. It is a state sequence consisting of position, velocity and heading of the target agent at each observation step, i.e., $I_{\text{hist}} = [(x_t, y_t, v_t^x, v_t^y, \sin\theta_t, \cos\theta_t)_{t=-T+1:0}]$. An agent-centric coordinate frame is defined by aligning the origin with the last observed position of a target agent and the x+ axis with its last observed heading, such that the target agent is located at the origin and heading to the east at the current time $t = 0$. The historical states of the target agent are then transformed to this coordinate frame. We repeat this coordinate frame normalization for all target agents to be predicted in a scene.

Agent Relation. This describes the relative states of other agents with respect to a target agent. Again, the historical states of other agents are transformed to the agent-centric coordinate frame of the target agent. We take into account the neighboring agents within a certain range (related to prediction horizon). Assuming B agents are considered, the agent relation is $I_{\text{rela}} = \{(I^i_{\text{rela}})_{i=1:B}\}$, where I^i_{rela}, similar to I_{hist}, is a sequence of position, velocity and heading of the i-th neighboring agent relative to the target agent.

Road Graph. This defines the map area where the target agent may reach in the prediction horizon. Similarly, the map elements are transformed to the agent-centric coordinate frame. We exploit lane centerlines, lane boundaries and traffic controls (e.g., turns and stop signs) to represent the road graph, and approximate each lane centerline or lane boundary by a polyline, which is an ordered set of connected line segments with equal length. A road graph with L polylines is represented by $I_{\text{road}} = \{(I^i_{\text{road}})_{i=1:L}\}$, where I^i_{road} is the i-th polyline. Let (a, b) be the endpoints of a segment, α be the heading angle, and c be the closest point on the segment to the origin. We represent a line segment by $[(a-b)/\|a-b\|_2, \|c\|_2, c/\|c\|_2, \|a-c\|_2, \alpha]$.

3.2. Agent-Centric Scene Representation (AcSR)

We refer to the collection of representation encoding of the heterogeneous input as AcSR. It provides a unified foundation for learning interactions across all input sources. We employ multilayer perceptron with gating (gMLP) [10], a simple yet effective method to model sequential data, to perform uniform encoding on various input sources.

Encoding Agent History. We adopt a gMLP G_{hist} to encode the historical states of a target agent as $F_{\text{hist}} = [f_{-T+1}, ..., f_0] = G_{\text{hist}}(I_{\text{hist}})$, where $F_{\text{hist}} \in \mathbb{R}^{T \times D_F}$ is a temporally ordered sequence of encoded features. We define F_{hist} as the full history encoding, and $\mathring{F}_{\text{hist}} = f_0$ as the compact history encoding that only contains the feature at the current time step $t = 0$. F_{hist} can be adopted as a memory to generate diverse proposals as it captures the complete spatio-temporal cues, while $\mathring{F}_{\text{hist}}$ can be thought of as a compact representation to constitute the set of AcSR.

Encoding Agent Relation. We use another gMLP G_{rela} to encode the relative states of neighboring agents with respect to a target agent as $F_{\text{rela}} = \{(F^i_{\text{rela}})_{i=1:B}\} = G_{\text{rela}}(I_{\text{rela}})$, where $F^i_{\text{rela}} \in R^{T \times D_F}$ is a temporally ordered sequence of encoded features of the i-th neighboring agent. All neighboring agents of a target agent share the same gMLP G_{rela}. Similarly, $\mathring{F}_{\text{rela}} = \{(\mathring{F}^i_{\text{rela}})_{i=1:B}\}$ is used to form the compact agent relation encoding for AcSR.

Encoding Road Graph. We utilize a third gMLP G_{road} to encode the polylines in the road graph around a target agent as $F_{\text{road}} = \{(F^i_{\text{road}})_{i=1:L}\} = G_{\text{road}}(I_{\text{road}})$, where $F^i_{\text{road}} \in \mathbb{R}^{T \times D_F}$ is an ordered sequence of encoded segments of the i-th polyline. All polylines in the road graph share the same gMLP G_{road}. We also collect the compact polyline encoding of a road graph to compose $\mathring{F}_{\text{road}} = \{(\mathring{F}^i_{\text{road}})_{i=1:L}\}$ as the local map representation in AcSR.

We combine the compact encoded features from the three aforementioned input sources to construct the AcSR, i.e., $S = \{\mathring{F}_{\text{hist}}, \mathring{F}_{\text{rela}}, \mathring{F}_{\text{road}}\}$. AcSR provides a single unified feature space for the complex interaction learning, leading to a succinct network architecture. Furthermore, the different input representations are uniformly encoded by gMLP, which is beneficial for reducing the gap between heterogeneous information and enables the following attention operations to better learn the interactions within the same source and across different sources alike.

3.3. Proposal Generation

To enhance the output multimodality, our network is designed to first learn to generate proposals merely from the motion history of a target agent. This maximizes the diversity of future trajectory candidates by waiving the constraints from the particular driving scenarios. We set the number of generated proposals to K, which is sufficiently larger than the required number of output modality M. A set of K learnable proposal queries Q_{prop} is applied to attend to the full history encoding F_{hist} through cross-attention to produce the set of proposals $E_{\text{prop}} = \{(E^i_{\text{prop}})_{i=1:K}\} \in \mathbb{R}^{K \times D_E}$, as demonstrated in Figure 2. It can be verified that, for the proposal generation, using $\mathring{F}_{\text{hist}}$ results in inferior performance than F_{hist}, which retains the full historical spatial and temporal information of the target agent (Section 4.6). To further encourage the diversity of proposals, Q_{prop} vectors are initialized orthogonally.

3.4. Anchor Learning

Anchors are learned end-to-end in the network to convey goal-oriented environmental information, while preserving diversity. We set the number of anchors to K, same as the number of proposals, so that each anchor corresponds to one proposal. We utilize self-attention on AcSR to let the heterogeneous elements in S attend to each other and learn related interactions with maximum flexibility. Assuming the output of self-attention is $\{(R_i)_{i=1:1+B+L}\}$, the target agent representation R_1 has fused the input context. Our network is then supervised to learn to predict K trajectory endpoints based on R_1. And the predicted endpoint that is closest to the ground truth is chosen as the correct one. A smooth ℓ_1 loss is optimized to minimize the distance between ground truth and correct prediction.

Our model does not directly adopt the predicted endpoints, but make use of their embeddings (i.e., the pre-output features) as the anchors $E_{\text{anch}} = \{(E^i_{\text{anch}})_{i=1:K}\} \in \mathbb{R}^{K \times D_E}$, which are utilized to inject the goal-oriented scene context to the generated proposals. This is a new genre of anchors compared to those used in previous works. In TNT [30], anchors are uniformly sampled from the map using hand-crafted rules. In MultiPath [1], anchors are a set of pre-defined trajectories clustered from training data. In MultiPath++ [21], anchors are learnable model parameters that are fixed after training and independent of input. As a contrast, we propose to exploit the anchor embeddings to facilitate trajectory learning. Compared to TNT and MultiPath, our anchors are obtained more adaptively and conveniently via end-to-end learning. Compared to MultiPath++, our anchors correspond to individual samples, thus carrying concrete sample-specific information.

3.5. Anchor-Informed Proposals (AiP)

As mentioned above, proposals can be viewed as unconstrained future trajectories inferred solely from agent history, and anchors convey goal based contextual information. Here we introduce AiP to enable the network for further selection and refinement, while achieving diverse output multimodality. One way to fuse proposals and anchors is through attention to use proposals to query anchors. However, we find this results in a collapse of multimodality to some extent. Instead, we directly sum proposals and anchors together to generate AiP, which is found to be simple

yet effective to accomplish the fusion in our experiments:

$$E_{\text{AiP}} = E_{\text{prop}} + E_{\text{anch}}, \tag{1}$$

where $E_{\text{AiP}} \in \mathbb{R}^{K \times D_E}$ is provided to the following hydra prediction heads to output the final predicted trajectories.

3.6. Hydra Prediction Heads

As discussed in Section 3.3, we intentionally generate more proposals than the required number of output modality (i.e., $K > M$). In practice, we set $K = 2M$, and randomly select M embeddings from E_{AiP} as a subset. This random selection is repeated N times, and thus N different subsets can be obtained. Accordingly, we construct N heads as the hydra prediction heads, each of which takes as input one subset and produces multimodal future trajectories. With the hydra heads and associated subsets that are randomly selected, the output modality can be further enhanced. In addition, this design also paves the way for ensembling by encouraging the hydra heads to learn complementary prediction. Our single network obtains a group of $N \times M$ trajectories via ensembling the predicted results from the hydra prediction heads. We then simply employ non-maximum suppression to merge them into the final prediction output with M multimodal trajectories.

We parameterize the distribution of each predicted trajectory as a Gaussian mixture model (GMM):

$$p(\tau) = \sum_{i=1}^{M} p^i \prod_{t=1}^{H} \mathcal{N}(\tau_t - \mu_t^i, \Sigma_t^i), \tag{2}$$

where τ is a trajectory in the prediction horizon H, p^i is the likelihood of the i-th mode, $\tau_t \in \mathbb{R}^2$ is the trajectory point at time step t with predicted mean $\mu_t^i \in \mathbb{R}^2$ and covariance $\Sigma_t^i \in \mathbb{R}^{2\times 2}$. For network training, we maximize the likelihood of ground truth under the predicted trajectory distribution. We apply hard labeling to assign the trajectory with the smallest endpoint distance to ground truth as positive. And only the errors between positive trajectories and ground truth are considered in the regression loss.

4. Experiments

In this section, we first describe our experimental setup including datasets, evaluation metrics and implementation details. A variety of ablation studies are conducted to understand the contribution of each individual design in our network. We report extensive comparisons with the state-of-the-art methods on two popular benchmarks.

4.1. Datasets and Metrics

We evaluate our approach on two large-scale motion forecasting datasets: Argoverse-1 [2] and Argoverse-2 [25]. **Argoverse-1** contains 333K real-world driving sequences selected from intersections or dense traffic. Each sequence consists of 2 seconds of history and 3 seconds of future, sampled at 10Hz. Following the official split, the training, validation and test sets have 205,942, 39,472 and 78,143 sequences respectively. **Argoverse-2** upgrades the first version on scenario complexity, prediction horizon and agent types. It contains 250K sequences mined for challenging scenes with rich geographic diversity. This new dataset extends the prediction horizon to 6 seconds, and covers 5 types of agents including vehicles, pedestrians, cyclists, motorcyclists and buses. There are 200,000, 25,000 and 25,000 sequences in training, validation and test sets.

Method	minADE_6	minFDE_6	minADE_1	minFDE_1
VectorNet [5]	-	-	1.66	3.67
LaneRCNN [28]	0.77	1.19	1.33	2.85
TPCN [26]	0.73	1.15	1.34	2.95
mmTransformer [12]	0.71	1.15	-	-
LaneGCN [9]	0.71	1.08	1.35	2.97
PAGA [4]	0.69	1.02	1.31	2.87
MultiPath [1]	0.80	1.68	-	-
TNT [30]	0.73	1.29	-	-
DenseTNT [7]	0.73	1.05	-	-
ProphNet-S	**0.68**	**0.97**	**1.28**	**2.77**

Table 1. Comparison of ProphNet and the state-of-the-art methods on the validation set of Argoverse-1. Groups 1 and 2 are the scene-centric and agent-centric methods.

We adopt the official evaluation metrics defined by the two benchmarks: minADE, minFDE, brier-minFDE, and MR. minADE (average displacement error) is the minimum of the time-average distance between predicted trajectories and the ground truth trajectory, over M prediction modes. minFDE (final displacement error) is the minimum distance between M predicted endpoints and the ground truth endpoint. brier-minFDE additionally takes the prediction probability into account. MR (miss rate) is the fraction of scenes where none of M predicted endpoints are within a certain distance to ground truth.

4.2. Implementation Details

We implement ProphNet in PyTorch [17] and train on 16 NVIDIA V100 GPUs with a batch size of 64. All gMLPs including G_{hist}, G_{rela} and G_{road} have the same hidden layer dimension 256. For all attention layers, the multi-head attention is configured to 4 heads, and the dimension of each head is 64. We employ one cross-attention layer for generating E_{prop} and three self-attention layers for E_{anch}. In all experiments, the numbers of output modality, AiP and hydra prediction heads are set as $M = 6$, $K = 12$ and $N = 3$, respectively. We use Adam [8] to train the model for 60 epochs, and set the initial learning rate to 2×10^{-4} and decay the learning rate to 2×10^{-5} after 50 epochs. We keep the same setting for both datasets.

Method	minADE_6	minFDE_6	minADE_1	minFDE_1	MR	brier-minFDE
LaneRCNN [28]	0.9038	1.4526	1.6852	3.6916	0.1232	2.1470
LaneGCN [9]	0.8703	1.3622	1.7019	3.7624	0.1620	2.0539
mmTransformer [12]	0.8436	1.3383	1.7737	4.0033	0.1540	2.0328
TPCN [26]	0.8153	1.2442	1.5752	3.4872	0.1333	1.9286
SceneTransformer [16]	0.8026	1.2321	1.8108	4.0551	0.1255	1.8868
TNT [30]	0.9097	1.4457	2.1740	4.9593	0.1656	2.1401
DenseTNT [7]	0.8817	1.2815	1.6791	3.6321	0.1258	1.9759
MultiPath++ [21]	0.7897	1.2144	1.6235	3.6141	0.1324	1.7932
Wayformer [15]	**0.7676**	1.1616	1.6360	3.6559	0.1186	1.7408
ProphNet	0.7726	**1.1442**	**1.5240**	**3.3341**	**0.1121**	**1.7323**

Table 2. Comparison of ProphNet (single model without ensembling) and the state-of-the-art methods on the test set of Argoverse-1. Groups 1 and 2 are the scene-centric and agent-centric methods.

Method	minADE_6	minFDE_6	minADE_1	minFDE_1	MR	brier-minFDE
GOHOME [6]	0.88	1.51	1.95	4.71	0.20	2.16
GoRela [3]	0.76	1.48	1.82	4.62	0.22	2.01
TENET [24]	0.70	1.38	1.84	**4.69**	0.19	1.90
ProphNet	**0.68**	**1.33**	**1.80**	4.74	**0.18**	**1.88**

Table 3. Comparison of ProphNet (single model without ensembling) and the state-of-the-art methods on the test set of Argoverse-2.

4.3. Results on Argoverse-1

We start from the comparison of ProphNet and the leading algorithms on the validation set of Argoverse-1. Here we do not use ensembling for a fair comparison against the methods without such a design. So we disable the hydra prediction heads and employ the single-head version, i.e., ProphNet-S. As demonstrated in Table 1, ProphNet-S outperforms other methods by a large margin. We attribute our superior result of minFDE to the proposed AiP, which fuses diversified proposals with goal-oriented anchors. And the anchor learning is explicitly trained to estimate the feasible goals, which is reflected in the more accurate endpoint prediction. This comparison clearly shows the advantage of the overall design of our approach.

We then evaluate ProphNet on the test set of Argoverse-1. We use dropout augmentation on agent relation and road graph, where 10% elements are randomly dropped. We compare with the results of published methods in Table 2. Compared to the leading scene-centric methods, ProphNet demonstrates strong advantage over all evaluation metrics. As for the comparison with most recent agent-centric methods, MultiPath++ [21] employs 5 heads and each produces 64 trajectories, while ProphNet uses 3 heads and each with 6 output trajectories. Wayformer [15] trains 15 models for ensembling. In contrast, our prediction result is generated by a single model. It is observed that ProphNet significantly outperforms MultiPath++, and overall performs better than Wayformer. In addition to improving the prediction accuracy, ProphNet is computationally efficient. See the details of latency comparison and analysis in Section 4.5.

4.4. Results on Argoverse-2

We also evaluate ProphNet on the test set of the latest benchmark Argoverse-2. We use the same configuration as that of Argoverse-1. Compared to the results of published methods, ProphNet achieves superior performance, in particular over TENET [24], which is the winner of the motion forecasting challenge on Argoverse-2 in 2022.

4.5. Latency Analysis

In addition to prediction accuracy, we take a closer look into inference latency, which is an equally important measurement for practical onboard deployment. We compare the inference latency of ProphNet with the state-of-the-art methods in Table 4. We report the inference latency on a single NVIDIA V100 GPU and set the number of agents to 64, a reasonable amount of neighboring agents in practice. For the scene-centric models, we first compute the average latency per agent. In general, the scene-centric models run considerably faster than most agent-centric models, often by an order of magnitude. In the contrary, as an agent-centric model, ProphNet achieves the inference latency around 28ms, significantly speeding up run time to the scene-centric level. We attribute our efficient inference to the uniform and succint model deisgn, as well

Method	Latency (ms)	GFLOPs
VectorNet [5]	10.9	0.01
LaneGCN [9]	63.4	0.13
mmTransformer [12]	14.2	0.01
DenseTNT [7]	490.1	0.62
MultiPath++ [21]	240.4	2.19
Wayformer [15]	102.3	2.13
ProphNet-S	27.4	0.39
ProphNet	28.0	0.40

Table 4. Comparison of ProphNet and its variant to the state-of-the-art methods on inference latency and number of GFLOPs. Groups 1 and 2 are the scene-centric and agent-centric methods.

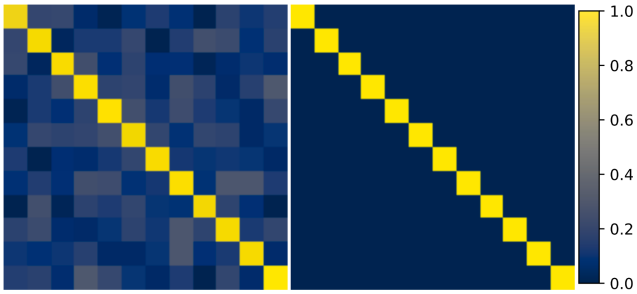

Figure 3. Illustration of the orthogonality of learnable proposal queries before (right) and after (left) training.

as ruling out the computationally heavy operations such as multi-axis [15, 16] and autoregressive sequential prediction [16, 24]. It is also observed that the hydra prediction heads in our approach only increase the latency slightly, from 27.4ms to 28.0ms.

4.6. Ablation Study

Next we conduct various ablation experiments to understand the contribution of each individual component in our design. As shown in Table 5, model (a) makes prediction using $\mathring{F}_{\text{hist}}$, and model (b) using E_{prop}. Since E_{prop} are generated based on F_{hist} that encodes the full historical states of a target agent, model (b) performs better than model (a). In model (c), the hierarchical encoding first applies $\mathring{F}_{\text{hist}}$ to attend to $\mathring{F}_{\text{rela}}$, and then attend to $\mathring{F}_{\text{road}}$. For model (d), the feature representation of a target agent in AcSR is directly used for prediction. Comparing models (c-d), we can see the advantage of a unified feature space for learning complex interactions over the modality-specific processing. By integrating E_{anch} into model (d), model (e) improves the result, in particular for minFDE, suggesting the merit of learned anchors for endpoint prediction. In comparison to model (f) using a single prediction head, the full model (g) with hydra prediction heads gains further improvement.

Model	minADE_1	minFDE_1
(a) Compact History Encoding	3.31	9.15
(b) Proposals	3.27	9.06
(c) Hierarchical Encoding	2.04	5.81
(d) AcSR	2.02	5.62
(e) AcSR with Anchors	2.01	5.47
(f) ProphNet-S	1.98	5.37
(g) ProphNet	1.97	5.33

Table 5. A variety of ablation studies and related design choices of ProphNet on the validation set of Argoverse-2.

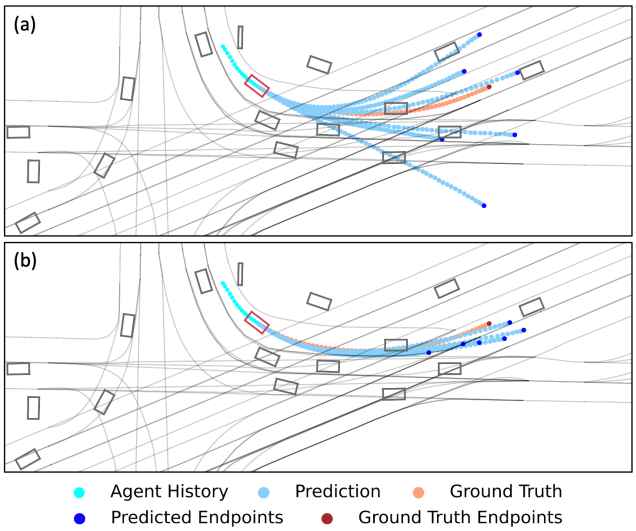

Figure 4. Illustration of the trajectories predicted (a) by the generated proposals directly and (b) by the anchor-informed proposals.

4.7. Qualitative Results

Queries. In Section 3.3, we introduce proposal queries that are orthogonally initialized for proposal generation to enhance output multimodality. As the queries are trainable, we investigate to what extent the learned queries can retain the orthogonality after training. As shown in Figure 3, the initial queries are orthogonal, so their off-diagonal similarities are zero. After training, we find the similarities of the off-diagonal elements are still close to zero. This indicates that our model well maintains the diversity to induce proposals, showing no signs of mode collapse at this level.

Proposals. As illustrated in Figure 4(a), the trajectories predicted directly from proposals fit well to the agent history from the perspective of kinematics, though they are freely diverse or even off-road. This is because in our design the proposals are merely generated based on the agent dynamics, without the contextual information such as agent interactions and map constraints. It is found that the proposals generated in this way can be fairly diversified, cater-

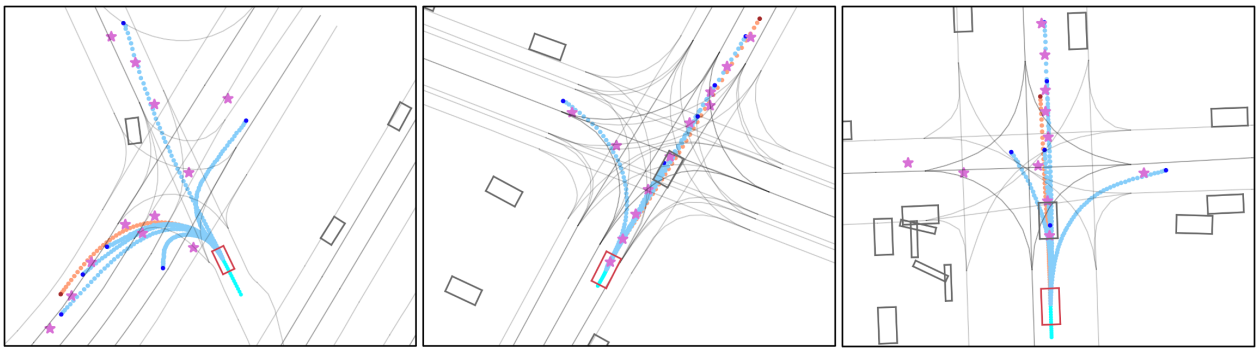

Figure 5. Illustration of the endpoints (pink stars) predicted by anchors in ProphNet. Note these endpoints are not used during inference, but are visualized here for better understanding of the learned anchors. The three examples also clearly demonstrate the diverse output multimodality enabled by our approach.

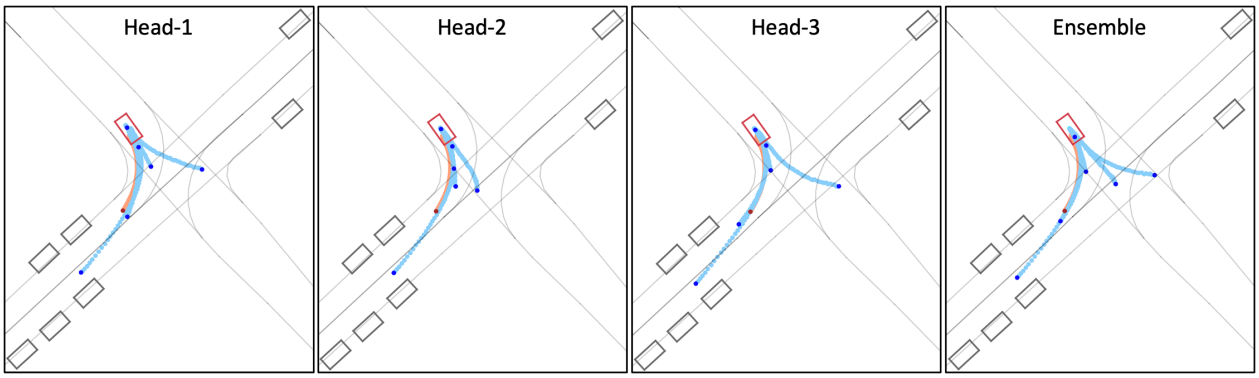

Figure 6. Illustration of the future trajectories output by hydra prediction heads and their ensemble in ProphNet.

ing to our purpose for rich coverage of potential motions at the early stage. When absorbing the goal-oriented scene context from anchors, the anchor-informed proposals produce accurate and multimodal future trajectories, as shown in Figure 4(b). The difference between (a-b) in this figure vividly presents the validity of the two components (i.e., proposals and anchors) in ProphNet.

Anchors. We make use of the supervision of endpoint estimation to guide the learning of anchors. Here we demonstrate the endpoints predicted from anchors in Figure 5. Although such predicted endpoints are not used during inference of our approach, we visualize them for in-depth understanding of what is embedded in the learned anchors. As illustrated in this figure, one can see that the predicted endpoints lie in road areas and mostly follow lane centerlines, which validates that the learned anchors have well encoded the scene context. Also, the predicted endpoints are positioned reasonably close to ground truth or the endpoints of other possible behavior modes, revealing the efficacy of anchor learning in our network.

Hydra Prediction Heads. Finally, we show each individual prediction result of the hydra heads in Figure 6. This is a challenging scenario as the target agent is static. We observe that the three prediction heads produce complementary diverse trajectories. After ensembling, the multimodal coverage of final output is further improved. These prediction results collectively verify our hydra-head design of randomly selecting subsets from AiP to enrich multimodality and encourage complementary prediction.

5. Conclusion

We have presented ProphNet, an agent-centric prediction model with anchor-informed proposals for efficient multimodal motion forecasting. ProphNet involves uniform encoding of heterogeneous input to construct a unified feature representation space AcSR, generating proposals and anchors to form AiP, and feeding AiP into hydra prediction heads to produce the multimodal output trajectories. Extensive experimental results on the two popular benchmarks demonstrate the effectiveness of our approach in terms of both prediction accuracy and inference latency. We hope this work would encourage more research toward practical network designs considered for real-world driving deployment, with not only high accuracy but also succinct architecture and efficient inference.

References

[1] Yuning Chai, Benjamin Sapp, Mayank Bansal, and Dragomir Anguelov. MultiPath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. In *CoRL*, 2019. 2, 3, 4, 5

[2] Ming-Fang Chang, John W Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, and James Hays. Argoverse: 3D tracking and forecasting with rich maps. In *CVPR*, 2019. 5

[3] Alexander Cui, Sergio Casas, Kelvin Wong, Simon Suo, and Raquel Urtasun. GoRela: Go relative for viewpoint-invariant motion forecasting. *arXiv:2211.02545*, 2022. 6

[4] Fang Da and Yu Zhang. Path-aware graph attention for HD maps in motion prediction. In *ICRA*, 2022. 2, 5

[5] Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. VectorNet: Encoding HD maps and agent dynamics from vectorized representation. In *CVPR*, 2020. 2, 5, 7

[6] Thomas Gilles, Stefano Sabatini, Dzmitry Tsishkou, Bogdan Stanciulescu, and Fabien Moutarde. GOHOME: Graph-oriented heatmap output for future motion estimation. In *ICRA*, 2021. 6

[7] Junru Gu, Chen Sun, and Hang Zhao. DenseTNT: End-to-end trajectory prediction from dense goal sets. In *ICCV*, 2021. 1, 2, 3, 5, 6, 7

[8] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *ICLR*, 2015. 5

[9] Ming Liang, Bin Yang, Rui Hu, Yun Chen, Renjie Liao, Song Feng, and Raquel Urtasun. Learning lane graph representations for motion forecasting. In *ECCV*, 2020. 1, 2, 5, 6, 7

[10] Hanxiao Liu, Zihang Dai, David R. So, and Quoc V. Le. Pay attention to MLPs. In *NeurIPS*, 2021. 4

[11] Rong Liu, Jinling Wang, and Bingqi Zhang. High definition map for automated driving: Overview and analysis. *The Journal of Navigation*, 2020. 1

[12] Yicheng Liu, Jinghuai Zhang, Liangji Fang, Qinhong Jiang, and Bolei Zhou. Multimodal motion prediction with stacked transformers. In *CVPR*, 2021. 2, 3, 5, 6, 7

[13] Chenxu Luo, Xiaodong Yang, and Alan Yuille. Exploring simple 3D multi-object tracking for autonomous driving. In *ICCV*, 2021. 1

[14] Chenxu Luo, Xiaodong Yang, and Alan Yuille. Self-supervised pillar motion learning for autonomous driving. In *CVPR*, 2021. 1

[15] Nigamaa Nayakanti, Rami Al-Rfou, Aurick Zhou, Kratarth Goel, Khaled S. Refaat, and Benjamin Sapp. Wayformer: Motion forecasting via simple and efficient attention networks. *arXiv:2207.05844*, 2022. 2, 6, 7

[16] Jiquan Ngiam, Benjamin Caine, Vijay Vasudevan, Zhengdong Zhang, Hao-Tien Lewis Chiang, Jeffrey Ling, Rebecca Roelofs, Alex Bewley, Chenxi Liu, Ashish Venugopal, David Weiss, Benjamin Sapp, Zhifeng Chen, and Jonathon Shlens. Scene Transformer: A unified multi-task model for behavior prediction and planning. In *ICLR*, 2021. 2, 6, 7

[17] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 5

[18] Tung Phan-Minh, Elena Corina Grigore, Freddy A. Boulton, Oscar Beijbom, and Eric M. Wolff. CoverNet: Multimodal behavior prediction using trajectory sets. In *CVPR*, 2019. 2

[19] Haoran Song, Di Luan, Wenchao Ding, Michael Yu Wang, and Qifeng Chen. Learning to predict vehicle trajectories with model-based planning. In *CoRL*, 2021. 3

[20] Yichuan Charlie Tang and Ruslan Salakhutdinov. Multiple futures prediction. In *NeurIPS*, 2019. 2

[21] Balakrishnan Varadarajan, Ahmed Hefny, Avikalp Srivastava, Khaled Refaat, Nigamaa Nayakanti, Andre Cornman, Kan Chen, Bertrand Douillard, Chi-Pang Lam, Dragomir Anguelov, and Benjamin Sapp. MultiPath++: Efficient information fusion and trajectory aggregation for behavior prediction. *arXiv:2111.14973*, 2021. 1, 2, 3, 4, 6, 7

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2

[23] Danfeng Wang, Xin Ma, and Xiaodong Yang. TL-GAN: Improving traffic light recognition via data synthesis for autonomous driving. *arXiv:2203.15006*, 2022. 1

[24] Yuting Wang, Hangning Zhou, Zhigang Zhang, Chen Feng, Huadong Lin, Chaofei Gao, Yizhi Tang, Zhenting Zhao, Shiyu Zhang, Jie Guo, Xuefeng Wang, Ziyao Xu, and Chi Zhang. TENET: Transformer encoding network for effective temporal flow on motion prediction. *arXiv:2207.00170*, 2022. 6, 7

[25] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting. In *NeurIPS*, 2021. 5

[26] Maosheng Ye, Tongyi Cao, and Qifeng Chen. TPCN: Temporal point cloud networks for motion forecasting. In *CVPR*, 2021. 5, 6

[27] Maosheng Ye, Jiamiao Xu, Xunnong Xu, Tongyi Cao, and Qifeng Chen. DCMS: Motion forecasting with dual consistency and multi-pseudo-target supervision. *arXiv:2204.05859*, 2022. 2

[28] Wenyuan Zeng, Ming Liang, Renjie Liao, and Raquel Urtasun. LaneRCNN: Distributed representations for graph-centric motion forecasting. In *IROS*, 2021. 5, 6

[29] Chen Zhang, Honglin Sun, Chen Chen, and Yandong Guo. BANet: Motion forecasting with boundary aware network. *arXiv:2206.07934*, 2022. 2

[30] Hang Zhao, Jiyang Gao, Tian Lan, Chen Sun, Benjamin Sapp, Balakrishnan Varadarajan, Yue Shen, Yi Shen, Yuning Chai, Cordelia Schmid, Congcong Li, and Dragomir Anguelov. TNT: Target-driven trajectory prediction. In *CoRL*, 2020. 1, 2, 3, 4, 5, 6