Hybrid Motion Representation Learning for Prediction from Raw Sensor Data

Depu Meng, *Graduate Student Member, IEEE,* Changqian Yu, Jiajun Deng, Deheng Qian, Houqiang Li, *Fellow, IEEE,* Dongchun Ren

***Abstract*—Motion prediction from raw LiDAR sensor data has drawn increasing attention and led to a surge of studies following two main paradigms. One paradigm is global motion paradigm, which simultaneously detects objects from point clouds and predicts the trajectories of each object in the future. The other paradigm is local motion paradigm, which directly performs dense motion prediction pointwisely. We observe that global motion prediction can benefit from local motion representation, since it contains rich local displacement contexts that are not explicitly exploited in global motion prediction. Correspondingly, local motion prediction can benefit from global motion representation, since it provides object contexts to improve prediction consistency inside an object. However, the complement of these two motion representations has not fully explored in the literature. To this end, we propose Hybrid Motion Representation Learning (HyMo), a unified framework to address the problem of motion prediction by making the best of both global and local motion cues. We have conducted extensive experiments on nuScenes dataset. The experimental results demonstrate that the learned hybrid motion representation achieves state-of-the-art performance on both global and local motion prediction tasks.**

***Index Terms*—motion prediction, joint perception and prediction, autonomous driving**

I. Introduction

MOTION prediction in modern autonomous driving system aims at forecasting the future positions of surrounding objects (including vehicles, pedestrians, and cyclists) in traffic scenes. There are two popular frameworks for motion prediction: motion prediction on top of perception results [2], [6], [16], [17], [27], [39], [77], and motion prediction from raw sensor data [7], [8], [36], [40], [47], [55], [72], [73]. Motion prediction on top of perception results typically follows a detection + prediction pipeline. The input of the prediction module is object trajectories in the past and predict the object trajectories in the future. Motion prediction from raw sensor data directly use raw sensor data (e.g., LiDAR point cloud) as input and predict future trajectories. This framework simplifies perception and prediction pipeline and may enjoy lower inference latency due to computation sharing. In this paper, we focus on motion prediction from raw LiDAR sensor data.

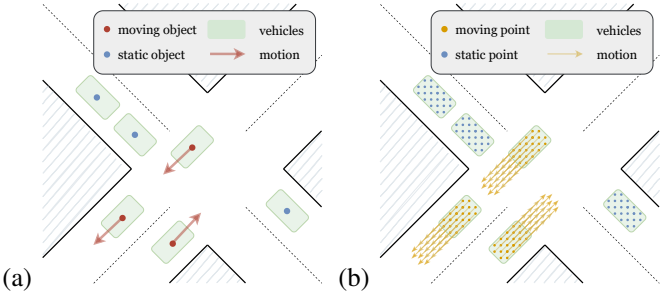

Fig. 1: Illustrations of **(a)** global motion and **(b)** local motion for vehicles. Global motion refers to the overall motion of objects. Local motion refers to motion of each point.

There are two main-stream paradigms for motion prediction from raw sensor data. One paradigm jointly detects objects in the point clouds, and predicts the future trajectories of object centers. The other paradigm predicts the motion of each point (or voxel) in the point cloud, without producing instance-level motion predictions.

Several attempts [7], [40], [47], [73] were made following the first paradigm. These methods model objects to 2D or 3D boxes and predict the trajectories of box center points in the future. In this paradigm, the predicted motion depicts the movement of objects (we call it *global motion*, as illustrated in Figure 1a). Thus, the motion is bonded with the instance and motion prediction relies on instance detection. This paradigm suffers from two downsides. The first downside is that, since in global motion paradigm, the motion prediction is coupled with object detection, the unstable label assignment caused by inaccurate object detection can affect the motion prediction supervision signal. The second downside is that, local displacement information (motion of each point or voxel) is not explicitly exploited in this paradigm. For instance, the information that background points (such as buildings, roads, and vegetation) are static is not included in this paradigm. While we observe that local displacement information can give useful context for global motion prediction.

The second paradigm [72], which means direct prediction of the motion of each point in the point clouds (we call it *local motion* , as illustrated in Figure 1b). Local motion is a finer portrayal of how elements move in the surrounding

Changqian Yu is the corresponding author.

Depu Meng is with Department of Automation, University of Science and Technology of China, Hefei, China (e-mail: mdp@mail.ustc.edu.cn). This work was done when Depu Meng was an intern at Meituan, Beijing, China.

Changqiang Yu, Deheng Qian, Dongchun Ren are with Meituan, Beijing, China (e-mail: changqianyu@meituan.com, qiandeheng@meituan.com, rendongchun@meituan.com).

Jiajun Deng and Houqiang Li are with the CAS Key Laboratory of Technology in Geo-Spatial Information Processing and Application System, Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei, China (e-mail: dengjj@ustc.edu.cn, lihq@ustc.edu.cn).

This work of Dr. Houqiang Li was supported by NSFC under contract No. 62021001.

environment. Despite its fine portrayal, motion prediction of dense points might suffer from noisy prediction problem (e.g., false alarms in background points). Further, dense local motion prediction result is not convenient to use in downstream tasks like motion planning.

Our approach is motivated by the observation that global motion and local motion representations are complementary to each other. Local motion representations can provide rich local contexts for global motion prediction. Prior knowledge like "most background stuff (buildings, roads, vegetation) are static" is not exploited in global motion paradigm, while can be discovered in local motion paradigm. Correspondingly, local motion paradigm lacks object contextual representations. Thus, the risk of inconsistent prediction within the same object, or in the background is higher.

We unify the two paradigms and present a Hybrid Motion Representation Learning (HyMo) approach, that jointly perform global motion prediction as well as local motion prediction. Our framework contains a 3D pyramid backbone for feature extraction, and two branches: local branch that contains two Conv-based heads for pixel classification and local motion prediction; global branch that contains a transformer decoder that aggregate features for instance-level prediction and the following classification, regression, and prediction heads for center detection and motion prediction. To evaluate the quality of learned hybrid motion representations, we establish an evaluation metric system that jointly evaluate the performance on global motion prediction, local motion prediction, object center detection, as well as pixel-level classification. Specifically, for local motion prediction, we evaluate Average Displacement Error (ADE) and Final Displancement Error (FDE) for both foreground (vehicle) points and background points. For global motion prediction, we first select true-positive detections, and then evaluate ADE/FDE on true-position detections.

We validate our HyMo approach on nuScenes [3] dataset. The experiment results demonstrate that the learned hybrid representations performs satisfactorily on both local motion prediction task and global motion prediction task. Further, object center detection quality is also boosted with the help of local representations.

The main contributions of our work include:

- We analyze global motion paradigm and local motion paradigm, and propose a unified paradigm: hybrid motion representation learning, for global motion prediction and local motion prediction.
- We study the relationship between global motion and local motion, and investigate how global motion prediction and object center detection can benefit from local supervision including local motion and pixel-wise classification.
- We build a transformer-based model that jointly predicts global motion, local motion, as well as detects object centers. The model is trained with an end-to-end fashion.
- We conduct experiments on both global motion prediction and local motion prediction tasks and setup an evaluation metric system to jointly evaluate the performance on global motion prediction, local motion prediction, object center detection and pixel classification tasks. Empirical studies on nuScenes are performed to validate the effectiveness of our proposed framework.

II. Related Work

Motion prediction. Motion prediction aims at predicting the position of surrounding objects in the future. Traditional motion prediction methods use object trajectories in the past and semantic map information to predict the future trajectories of objects. A motion prediction model commonly consist of an encoder module that encodes object trajectories along with semantic map to latent representations, and a motion prediction head to perform the trajectory prediction. Many works [11], [17], [25], [26], [38], [39], [49], [77] focus on learning map and object trajectory representations. As for the input encoding, there are three popular types: rasterization [7], [9], [12], [19], [54], graph encoding [17], [21], [39], [79], [83], and point-cloud encoding [23], [77]. Aside from input representation, many works focus on design of prediction head architecture. Single-shot methods [17], [39] directly regress the future trajectories of objects. Recently, goal-based motion prediction methods [19], [21], [48], [60]–[62], [83] achieve superior performance compared to single-shot methods both on vehicle motion prediction and pedestrian motion prediction. Goal-based methods first predict the end-point of the object's future trajectories, then predict the whole trajectories conditioned on the goal position. In our paper, we do not use object trajectories or semantic maps as input. We directly extract object positions as well as their motion representations from raw LiDAR point cloud data. Apart from model architecture design, [1], [22], [63], [67], [68] focus on interactions between different agents and between agents and maps. Since past trajectories are served as input, these methods rely on accurate object detection and tracking.

Motion prediction from raw sensor data. Another direction is motion prediction from raw sensor data. Motion prediction from raw sensor data performs object detection as well as motion prediction simultaneously. FaF [47], IntentNet [7], HDNet [73] jointly detect objects and predict their future trajectories. PnPNet [40] introduces a tracking module and exploits trajectory features for motion prediction. MP3 [8] jointly performs motion planning as well as perception and prediction. MotionNet [72] provide a new formulation for motion prediction. Instead of predicting motion of each object, MotionNet predicts motion of each pixel, which can serve as a backup as bounding-box-based methods. HyMo jointly learns a hybrid motion representation that both for instance-wise motion prediction and pixel-wise motion prediction.

Transformer-based 2D object detection. Previously, popular 2D object detection methods [4], [20], [28], [31], [33]–[35], [37], [41], [42], [45], [57]–[59], [69], [78], [82], [84], [86], [87] adopt CNNs to construct object detection head. DETR [5] proposes a transformer-based object detection head and demonstrated its simplicity and effectiveness compared to CNN-based methods. DETR removes two hand-craft parts in popular object detection frameworks: Non-Maximum Suppression (NMS) and anchor generator. However, DETR suffers from slow convergence issue. Many works are proposed to

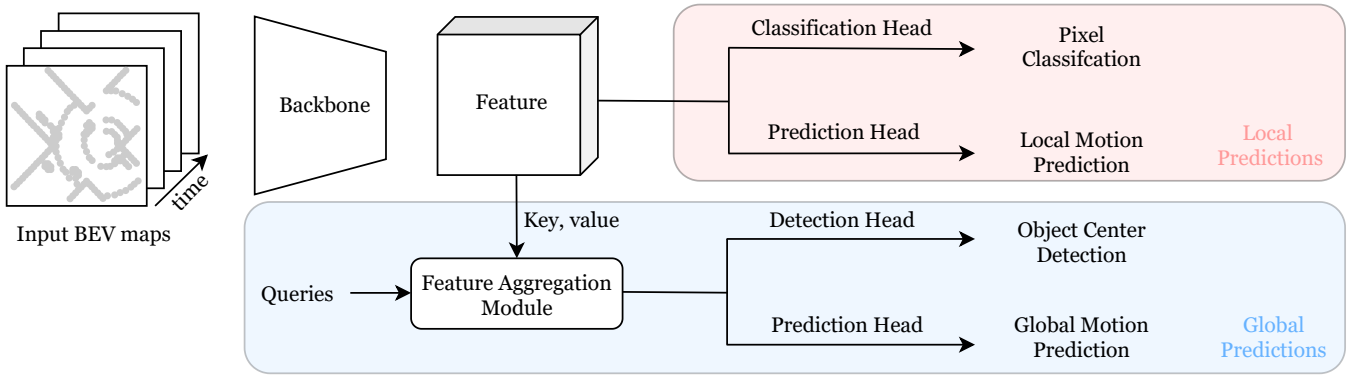

Fig. 2: An illustration of the HyMo framework. HyMo consists of a backbone for feature extraction, local prediction heads, and global prediction heads. Global representations are aggregated from local representations through a Feature Aggregation Module.

accelerate the training process [13], [18], [50], [81], [88]. In our method, we follow DETR framework to perform global predictions. The features for local predictions are used as keys and values in decoder cross-attention. Thus, the decoder aggregates local prediction information to make global predictions.

3D object detection. Several methods try to detect 3D objects from point clouds [14], [15], [29], [32], [51], [56], [64]–[66], [74], [76], [85] or from raw images [43], [71], [75]. Recently, transformer [70]-based 3D object detection became a popular trend. GroupFree [44] computes the feature of an object from all the points in the point cloud with an attention mechanism. 3DETR [52] gets rid of CNN-based backbone and introduce a pure transformer-based framework. We focus on motion prediction, and only detect object center points which serves as the starting point of future trajectory, instead of predicting the object bounding box. Following [5], [44], [50], [52], [89], we adopt a transformer decoder in object center detection and bipartite matching technique in label assignment.

III. Methodology

A. Overall Framework

Pipeline. As illustrated in Figure 2, the proposed framework follows a standard multi-tasking pipeline: a backbone for feature extraction and multiple prediction heads for different tasks. The LiDAR point clouds are converted to bird-eye-view (BEV) maps . 5 consecutive frames of BEV maps serve as the input of our model. For backbone, we use the same architecture as [72], which outputs a 2D feature map. For pixel-wise motion prediction (local motion prediction) and classification, we use a light-weight convolution head. For object-level motion prediction (global motion prediction) and object center detection, we use a transformer decoder layer for feature aggregation. MLPs on top of the decoder are used for object classification, center regression, and motion prediction.

Motion prediction. Following [72], we predict the position offset $\mathbf{o}_i$ between two consecutive frames i and $i-1$. The total displacement from the reference frame to frame t is computed by a summation: $\mathbf{m}_{(0)}^{(t)} = \sum_{\tau=1}^{t} \mathbf{o}_\tau$. We predict the displacement in the future 1s. There are 20 frames in total, and the interval between two consecutive frames is 0.05s. For local motion prediction, we predict an additional binary static/moving state similar to [72] to suppress noise predictions in the background.

Object center regression and classification. We follow the DETR [5] framework for object center regression and classification. For center regression, a candidate center is predicted from the decoder output embedding

$$\mathbf{c} = \mathrm{sigmoid}(\mathrm{MLP}(\mathbf{f}) + \mathbf{r}) \tag{1}$$

Here $\mathbf{c}$ is a 2D point (x, y) which represents the predicted object center. $\mathrm{sigmoid}$ is used to normalize the center position to $[0, 1]$. $\mathbf{f}$ is the decoder output embedding, MLP is the center regression head and $\mathbf{r}$ is a learned 2D point that serves as the initial object center of the query, so that the network predicts the offset between $\mathbf{c}$ and $\mathbf{r}$ using feature $\mathbf{f}$. $\mathrm{sigmoid}$ is used for normalization.

For object classification, the classification score for each candidate center is also predicted from the decoder output embedding $\mathbf{f}$ through an MLP.

Pixel-wise classification. We divide pixels into two categories: foreground category and background category. All pixels belong to the vehicle category are defined as foreground category, and other pixels are defined as background category. A 2-layer convolution head is used to predict pixel-wise classification.

B. Hybrid Motion Representation Learning

The proposed Hybrid Motion Representation Learning is a multi-task learning framework. The feature map extracted from the backbone is used to perform dense local motion prediction, as well as fed into a transformer decoder module for feature aggregation. The aggregated feature is used for global motion prediction and object center detection. The architecture of global prediction branch is illustrated in Figure 3.

Global motion representation learning. For global motion prediction, we first aggregate features into K candidate feature embeddings. Then each candidate embedding is used to predict a classification score, a center position, and a future trajectory. For label assignment, we adopt bipartite matching technique and assign one candidate object for each ground-truth object.

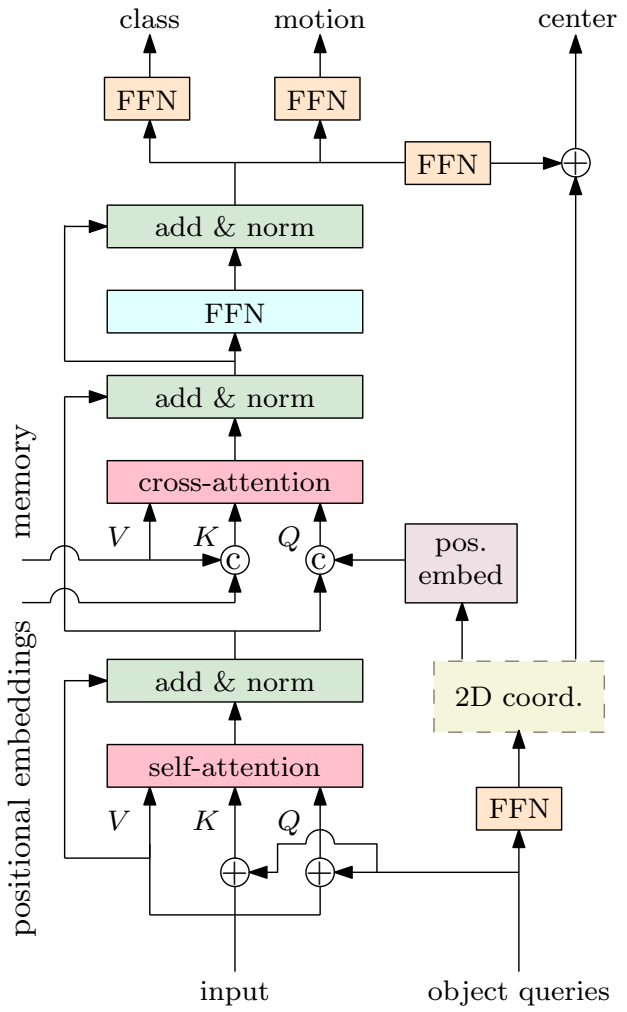

Fig. 3: An illustration of the global prediction branch of HyMo. The global prediction branch consists of a transformer decoder and a classification head, a motion prediction head, and a center regression head. 2D coordinates $\mathbf{r}$ are generated by object queries to serve as the initial prediction centers.

For feature aggregation, a transformer decoder layer is adopted. A transformer decoder consists of a self-attention layer, a cross-attention layer, and a feed-forward network. The self-attention layer provides interaction between queries. For cross-attention, we follow [50] to decouple the spatial attention weight and content attention weight in cross-attention. First, we mask the region of feature map output by backbone that does not contain LiDAR points. The masked feature map serves as cross-attention values. The keys are the addition of the masked feature map and its sinusoidal positional embeddings. For queries, we follow DETR to adopt N learned positional embeddings as decoder queries. Reference points are predicted from decoder query embeddings $\mathbf{q}$: $\mathbf{r} = \mathrm{Linear}(\mathbf{q})$. We map the reference points into sinusoidal positional embeddings , which serve as decoder queries. We set the default number of queries to 100.

For label assignment, we use a bipartite matching algorithm that assigns each ground-truth object to one candidate prediction, similar to DETR. The matching cost between prediction $\mathbf{y}$ and ground-truth $\hat{\mathbf{y}}$ consists of two components: classification cost and center regression cost:

$$\ell_{\text{match}} = \ell_{\text{cls}} + \ell_{\text{reg}} \tag{2}$$

Here ℓ_{match} is the overall cost function for matching. ℓ_{cls} and ℓ_{reg} represent classification and regression cost respectively. For classification cost, we use binary focal loss with $\alpha = 0.25, \gamma = 2.0$. Our regression cost is defined as L1 loss between the predicted center and ground-truth center. The classification cost filters out the high confidence predictions and the regression cost forces the position of predicted center and ground-truth center to be close. Here we do not consider motion prediction cost in matching, since the motion prediction might not distinct between objects (e.g., many objects are static, thus they have the same motion).

For loss function, we use a combination of classification, object center regression, and motion regression losses. We use the same classification and regression loss in matching. For motion prediction,

$$\ell_{\text{global}} = w_{\text{cls}}^{(g)}\ell_{\text{cls}} + w_{\text{reg}}^{(g)}\ell_{\text{reg}} + w_{\text{pred}}^{(g)}\ell_{\text{pred}} \tag{3}$$

Here ℓ_{pred} is a L1 regression loss on the displacement from the reference frame to each frame in the future.

$$\ell_{\text{pred}}(\mathbf{m}, \hat{\mathbf{m}}) = \sum_{t=1}^{T} \ell_1(\sum_{\tau=1}^{t} \mathbf{o}_\tau, \hat{\mathbf{m}}_{(0)}^{(t)}) \tag{4}$$

Here $\hat{\mathbf{m}}_{(0)}^{(t)}$ is the ground-truth displacement from the reference frame to frame t. $\mathbf{o}$ is the output of prediction head.

Local motion representation learning. Different from global motion, local motion annotation is usually difficult to acquire. We utilize two motion priors to generate pseudo local motion annotations from global motion annotations. The first prior is that most background points (e.g., buildings, roads, vegetation) are static. The second prior is that points belonging to the same object often have similar motion. With the first prior, we set motion of background points to be zero. Correspondingly, for points belonging to foreground objects, we generate two types of pseudo motions for foreground points. The first type directly regard global motion (motion of object center point) to be the local motion of points belonging to the same object.

$$\mathbf{p}_t = \mathbf{p}_0 + (\mathbf{c}_t - \mathbf{c}_0) \tag{5}$$

Here $\mathbf{p}_t$ is position of a point at time t. $\mathbf{c}_t$ means the position of object center at time t. we name this type of pseudo local motion to be *homogeneous* (Figure 4a). The other types regard objects as rigid boxes and compute motion of each point in the box by displacement and rotation of the box.

$$\mathbf{p}_t = \mathbf{R}^\top(\mathbf{p}_0 - \mathbf{c}) + (\mathbf{c}_t - \mathbf{c}_0) \tag{6}$$

Here $\mathbf{R}$ means the rotation transformation matrix of the box. We name this type of pseudo local motion to be *inhomogeneous* (Figure 4b).

For local motion supervision, we adopt L1 loss to regress the displacement from the reference frame to each future frame. Following [72], to suppress the noise prediction for the static background pixels, we also predict a binary static/moving state

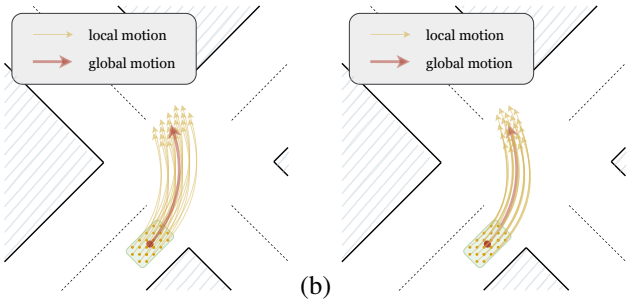

Fig. 4: Illustrations of **(a)** homogeneous local motion and **(b)** inhomogeneous local motion for vehicles. Homogeneous local motion of all points in an object is the same as object global motion. Inhomogeneous local motion for different points in an object can be different, since the rotation of objects are considered.

$\mathbf{s}$ for each pixel. The predicted motion is masked by the binary state:$\tilde{\mathbf{m}} = \mathbf{I}_{\mathbf{s}=\text{moving}} \cdot \mathbf{m}$

The overall loss function for local motion prediction, state prediction, and classification is formulated as

$$\ell_{\text{local}} = w^{(l)}_{\text{pred}}\ell_{\text{pred}} + w^{(l)}_{\text{state}}\ell_{\text{state}} + w^{(l)}_{\text{cls}}\ell_{\text{cls}} \tag{7}$$

Joint learning. Local motion representations contain rich local displacement contexts that is not exploited in global motion representations, and global motion representation contain object contexts that might improve the consistency of local motion prediction. Thus, we propose HyMo to jointly learn local motion representations and global motion representations. We combine the global motion loss function and local motion loss function together to form the overall loss function:

$$\ell = \ell_{\text{global}} + \ell_{\text{local}} \tag{8}$$

Connection to related methods. PnPNet [40] performs perception and motion prediction from raw sensor data. PnPNet focuses on developing an end-to-end model that jointly perform object detection, tracking, and motion prediction. Our approach focuses on building a framework that jointly perform local and global motion prediction. Settings of our approach and PnPNet are also different: (1) PnPNet does not perform local motion prediction while our approach does. (2) We do not utilize HD maps as input while PnPNet does. (3) PnPNet detects object boxes while we focus on motion prediction and only detect object centers.

MotionNet [72] performs local motion prediction and pixel classification. The local motion prediction module in our approach is similar to MotionNet, while simpler. We only adopt pixel classification loss, motion prediction loss and motion state loss, without the consistency losses in MotionNet. We also predict global motion that is not supported in MotionNet.

C. Network

For backbone, we use a spatial-temporal pyramid network proposed by [72]. It contains four spatio-temporal convolution (STC) blocks to reduce spatial and temporal resolution of the feature map, and generate a 5-level multi-scale feature pyramid: $T_1 \times C \times H \times W$, $T_2 \times 2C \times \frac{H}{2} \times \frac{W}{2}$, $T_3 \times 4C \times \frac{H}{4} \times \frac{W}{4}$, $T_4 \times 8C \times \frac{H}{8} \times \frac{W}{8}$, and $T_5 \times 16C \times \frac{H}{16} \times \frac{W}{16}$. $T_1 \geq T_2 \geq T_3 \geq T_4 \geq T_5$. A temporal pooling module is applied to each feature map to squeeze the temporal dimension to 1. Then, a series of de-convolutions is applied to recover the spatial resolution of the feature map to $H \times W$.

The local branch contains three prediction heads: pixel classification head, motion prediction head, and state (moving/static) estimation head. All three heads are formed by one Conv 3×3 followed by one Conv 1×1 layer.

The global branch contains one transformer decoder, and two prediction heads, as illustrated in Figure 3. The decoder consists of one self-attention layer, one cross-attention layer and one feed-forward network. The classification head, center regression head, and motion prediction head are consist of a three-layer MLP: $\text{Linear} \rightarrow \text{ReLU} \rightarrow \text{Linear} \rightarrow \text{ReLU} \rightarrow \text{Linear}$.

IV. Experiments

A. Dataset

nuScenes [3] is a large-scale autonomous driving dataset. It contains 1000 scenes, each scene is a 20 second log. Multiple types of sensor data are provided, including 32-beam 20 Hz LiDAR point clouds, RGB images, etc. In this paper, we only use the LiDAR point cloud data. Following [72], we use 500 scenes for training, 100 for validation and 250 for testing.

B. Evaluation Metrics

We build a set of evaluation metrics for evaluating our model on both instance-wise tasks and pixel-wise tasks. By default, we only concern about vehicles in the scene, and treat other objects as background.

Instance-wise evaluation. For instance-wise evaluation, we use center AP^C_θ to evaluate the vehicle center detection quality. d is the distance between the predicted vehicle center and ground-truth vehicle center, θ is the distance threshold, we regard prediction-gt pairs with $d \leq \theta$ as positive pairs. The predictions are sorted by the descending order of classification scores for AP computation. $\text{AP}^C_{2\text{m}}$, $\text{AP}^C_{5\text{m}}$, $\text{AR}^C_{2\text{m}}$, $\text{AR}^C_{5\text{m}}$, are reported. We evaluate the prediction quality for True Positive (TP) detections, which means matched predictions under $d \leq 2$ m. We evaluate all TP detections, as well as top 60% detections with highest classification score. Standard ADE and FDE metrics are used for motion prediction evaluation. We evaluate the trajectories of vehicles in the future 1 second. The ADE is the average displacement error over 4th, 8th, 12th, 16th, and 20th frame, corresponding to 0.2 s, 0.4 s, 0.6 s, 0.8 s, and 1.0 s in the future.

Pixel-wise evaluation. For pixel-wise evaluation, we evaluate the pixel-wise classification and pixel-wise prediction quality. The pixel-wise classification and prediction evaluation is only performed on pixels that contain one or more point clouds. We use class accuracy, mean class accuracy (MCA), and pixel accuracy as evaluation for classification, and use category pixel prediction ADE, FDE for prediction. We ignore pixels belonging to other foreground categories except vehicle for

TABLE I: Comparison of HyMo with other motion prediction methods on global motion prediction and local motion prediction task. HyMo performs better than other methods on global motion prediction. HyMo performs similarly to LocalMotion and MotionNet on local motion prediction. GlobalMotion+ means GlobalMotion trained with $2\times$ epochs.

Method	Global Motion Prediction				Local Motion Prediction			
	ADE@100%TP	FDE@100%TP	ADE@60%TP	FDE@60%TP	Vehicle ADE	Vehicle FDE	Bg ADE	Bg FDE
Constant Position	0.92 m	1.52 m	0.77 m	1.28 m	1.09 m	1.81 m	0.0 m	0.0 m
Constant Velocity	0.68 m	1.12 m	0.62 m	1.02 m	0.54 m	0.90 m	0.07 m	0.12 m
GlobalMotion	0.66 m	1.09 m	0.59 m	0.97 m	–	–	–	–
GlobalMotion+	0.66 m	1.08 m	0.44 m	0.73 m	–	–	–	–
FaFNet [47] (our-imple.)	0.73 m	1.19 m	0.58 m	0.96 m	–	–	–	–
LocalMotion	–	–	–	–	0.24 m	0.41 m	0.01 m	0.01 m
MotionNet [72]	–	–	–	–	0.22 m	0.38 m	0.01 m	0.01 m
HyMo	0.31 m	0.53 m	0.15 m	0.25 m	0.22 m	0.38 m	0.01 m	0.03 m

TABLE II: Comparison of HyMo with other methods on object center detection and pixel classification task. For object center detection, HyMo achieves significantly better AP than GlobalMotion and FaFNet. For pixel classification, HyMo performs better than LocalMotion and MotionNet for vehicle pixels. GlobalMotion+ means GlobalMotion trained with $2\times$ epochs.

Method	Object Center Detection				Pixel Classification			
	AP^C_{2m}	AP^C_{5m}	AR^C_{2m}	AR^C_{5m}	Vehicle Acc	Bg Acc	MCA	Pixel Acc
GlobalMotion	5.6	28.0	22.9	77.0	–	–	–	–
GlobalMotion+	15.7	52.1	34.8	84.9	–	–	–	–
FaFNet [47] (our-imple.)	38.7	49.0	64.4	92.0	–	–	–	–
LocalMotion	–	–	–	–	88.3	98.3	93.3	98.2
MotionNet [72]	–	–	–	–	90.3	98.7	94.4	98.2
HyMo	45.8	77.0	60.4	92.1	97.1	95.4	96.2	96.8

motion prediction. We predict the future trajectories in the future 1 second.

C. Implementation Details

Data pre-processing. Following [72], we crop the point clouds to a 64 m $\times$ 64 m $\times$ 5 m cube. The cube is then voxelized, and converted to a bird-eye-view (BEV) map with shape $256 \times 256 \times 13$. We concatenate 5 consecutive frames of synchronized point clouds together to form a 4D tensor as input. An empty mask is also formed to denote the pixels in the BEV map that do not contain LiDAR points.

Training. We use PyTorch 1.9 [53] to build and train our model. The models are trained from scratch with batch-size 16 on a single Tesla A100 GPU. We train the models using Adam [30] optimizer for 45 epochs with a multi-step learning rate schedule. The learning rate is reduced by half at $10, 20, 30$, and 40 epochs. The initial learning rate is set to 0.0016. We set weight decay to 0 in training.

Loss weights. We set object classification center regression and motion prediction loss balance weight $w^{(g)}_{\text{cls}}$, $w^{(g)}_{\text{reg}}$, $w^{(g)}_{\text{pred}}$, to 0.5, 0.5, 5, respectively. The pixel classification, prediction, and state estimation loss weight $w^{(l)}_{\text{cls}}$, $w^{(l)}_{\text{pred}}$, $w^{(l)}_{\text{state}}$, to 1, 1, 1, respectively. The overall loss for training is the summation of the above losses. Especially, to ease the class imbalance issue, we apply a background weight factor 0.05 in pixel classification and prediction. For bipartite matching, we set the balance factor for classification score and center distance to $1:1$.

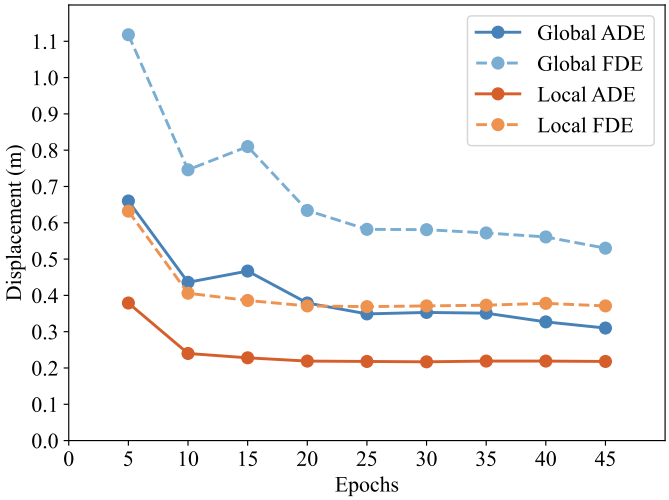

Fig. 5: Convergence curves of local motion prediction and global motion prediction in HyMo. Local motion prediction takes about 20 epochs to converge. ADE/FDE on validation set are reported. Global motion converges much slower than local motion prediction. Local motion prediction error is lower than global motion prediction error over the whole training process.

Motion prediction. Following [72], we predict the relative offsets between future frames, instead of directly precicting future object locations. For an object with current location $\mathbf{x}_0$ future locations $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n$, the motion prediction supervision of the model is $\tilde{\mathbf{x}}_i = \mathbf{x}_i - \mathbf{x}_{i-1}$.

TABLE III: Detailed motion prediction comparison on moving and static objects. Global FDEs are evaluated with 60% TP detections. **S.** refers to static objects, **M.** refers to moving objects, **O.** refers to overall objects. For global motion prediction, HyMo improves prediction on both static and moving objects. The improvement on moving objects is significant. For local motion prediction, the improvement is marginal.

Method	Global FDE (m)			Local FDE (m)		
	S.	M.	O.	S.	M.	O.
Constant Position	0.04	4.70	1.28	0.000	2.41	1.81
Constant Velocity	0.10	3.68	1.02	0.088	1.16	0.90
GlobalMotion	0.09	2.73	0.97	–	–	–
FaFNet [47] (our-imple.)	0.15	2.70	0.96	–	–	–
LocalMotion	–	–	–	0.008	0.55	0.41
MotionNet [72]	–	–	–	0.005	0.51	0.38
HyMo	0.06	0.83	0.25	0.005	0.50	0.38

D. Experiment Results

Comparison on Motion Prediction. We compare our proposed HyMo approach to other motion prediction methods in Table I. Both global motion prediction and local motion prediction are evaluated. We evaluate two *constant baselines*: constant position and constant velocity on nuScenes dataset. Constant position refers to the model that predicts all objects to be static. Constant velocity refers to the model that assumes all the objects move with constant velocity and regress the object velocities. Constant position baseline performs the worst. Constant velocity baseline performs much better than constant position. For both global and local motion prediction, HyMo significantly outperforms constant position and constant velocity models.

When compare to FaFNet [47], HyMo outperforms FaFNet by a large margin (1.19 m FDE@100%TP vs. 0.53 m FDE@100%TP). FaFNet performs similarly to the GlobalMotion baseline. Both of them only perform global motion prediction and suffer from slow convergence.

We also build a Global Motion Baseline (GlobalMotion), which means only performs global motion representation learning as described in paragraph "**Global motion representation learning**" in Sec. III-B.

For global motion prediction, we both evaluate prediction displacement on 100% True Positive (TP)s, and the top 60% TPs with highest classification score. As shown in Table I, when compared to GlobalMotion, our approach reduces a half of displacement error when evaluated on 100% TPs (ADE: 0.31 m vs. 0.66 m, FDE: 0.53 m vs. 1.09 m), and reduces 3/4 of displacement error when evaluated on 60% TPs (ADE: 0.15 m vs. 0.59 m, FDE: 0.25 m vs. 0.97 m).

Table I also provides a comparison between our proposed approach and Local Motion Baseline (LocalMotion), which is described in paragraph "**Local motion representation learning**" in Sec. III-B. We evaluate our approach and LocalMotion on the local motion prediction task. Our proposed approach performs better than LocalMotion for vehicle pixels (0.38 m vs. 0.41 m FDE). For background pixels, the motion prediction performance of our proposed approach and LocalMotion are similar. HyMo performs similarly to MotionNet [72] on the local motion prediction task.

Comparison on Perception Tasks. Table II provides a comparison on perception tasks, including object center detection and pixel classification. On object center detection, our approach significantly outperforms GlobalMotion. HyMo achieves 45.8 AP, while GlobalMotion only achieves 5.6 AP. Compared to FaFNet [47], the detection AP is improved by a large margin while the AR is similar.

We suspect that the GlobalMotion suffers from slow training convergence, so we try to double the training schedule of GlobalMotion and resulting in GlobalMotion+. GlobalMotion+ has better detection performance than GlobalMotion, while still much inferior to our proposed approach. For motion prediction, GlobalMotion+ does not perform notably better than GlobalMotion. This implies that the reason of poor prediction quality of GlobalMotion is not slow convergence but lack local motion contexts.

For pixel classification, HyMo has much higher accuracy for vehicle class (97.1 vs.88.3) and lower accuracy on background class (95.4 vs. 98.3). LocalMotion has a much lower vehicle accuracy than background, which implies that many vehicle pixels are misclassified into background category. MotionNet [72] performs better than LocalMotion, while HyMo still outperforms MotionNet classification on vehicle pixels by a large margin. Object context in HyMo make the model to be more focus on object region, and results in a more class-balanced classification.

Analysis of motion prediction quality for moving and static objects. Table III shows the motion prediction comparison on both static and moving objects. For both global motion prediction and local motion prediction, the prediction error on moving objects are significantly larger than the prediction error on static objects. For global motion prediction, For HyMo, both static and moving objects' motion prediction is improved. The improvement on moving objects is more significant. For local motion prediction, the improvement of HyMo on both static and moving objects is marginal.

Convergence curves of local motion and global motion prediction. Figure 5 shows the convergence curves of local motion prediction and global motion prediction in HyMo. Local motion prediction converges faster and has a lower displacement error consistently. This implies that local motion representations are able to provide accurate local motion context to global motion prediction in the early stage of training.

On the contrary, global motion prediction relies on object center detection. In early training, when object center detection is not learned well, the learning of global motion prediction will be adversely affected.

Running time. We evaluate the running time of the proposed HyMo model. The hardware we use is a PC with a Intel Core i7-12700K CPU, NVIDIA RTX 3090 GPU, 64 Gib memory. The software configurations are: Ubuntu 20.04 OS, CUDA 11.3, and PyTorch 1.9. We run HyMo on 1000 frames, and the average latency is 34 ms, and FPS 29.29.

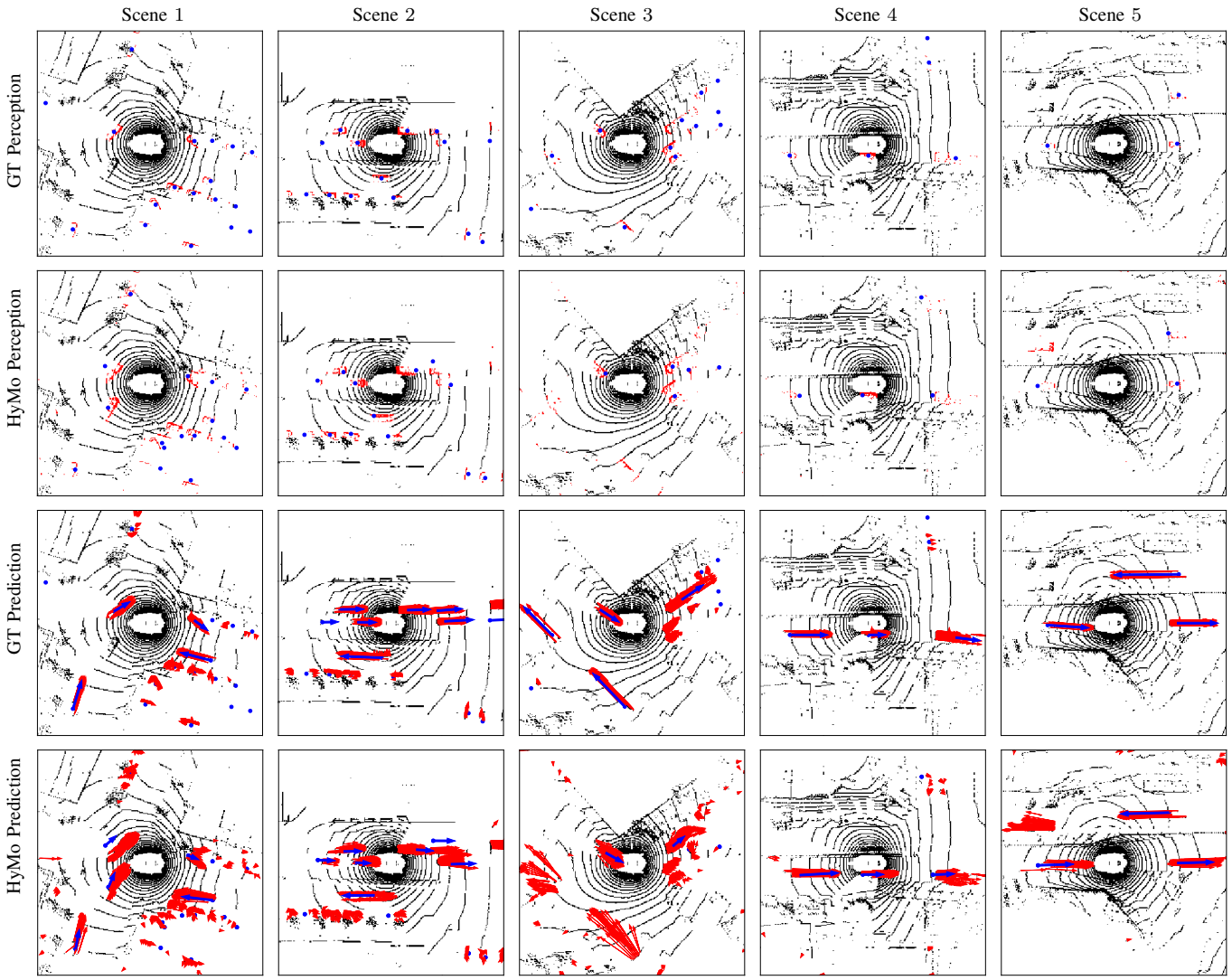

Fig. 6: Visualizations of HyMo predictions. GT Perception refers to ground-truth pixel classification map (red pixels for vehicles) and object centers (blue dots). HyMo Perception refers to perception results from HyMo. GT Prediction refers to ground-truth local (red arrows) and global motion (blue arrows). HyMo Prediction refers to predicted local and global motion.

TABLE IV: Ablation study on how local supervisions affect object center detection and global motion prediction. Local motion significantly boosts object center detection and global motion prediction performance. Pixel classification further improves object center detection performance on top of local motion.

Method	Local Motion	Pixel Cls.	AP^{C}_{2m}	FDE (m)
GlobalMotion			5.6	1.09
	✓		37.6	0.58
HyMo	✓	✓	45.8	0.53

E. Ablation Studies

Ablation study: local supervisions. In our approach, two types of local supervisions are introduced: local motion prediction and pixel classification. We decouple the two local supervisions and perform experiments in Table IV.

Firstly, we add a local motion supervision to the GlobalMotion baseline. Both the object center detection quality and global motion prediction quality are boosted. The FDE (in ablation study, we report FDE@100%TP by default) is reduced from 1.09 m to 0.58 m, which is close to HyMo (0.53 m). This indicates that local motion representations can help both object center detection and global motion prediction.

If we then add a pixel classification supervision on top of the local motion supervision, the object center detection is further improved by $+8.2$ AP^{C}_{2m}. Meanwhile, the global motion prediction quality only improves by a small margin 0.05 m. This implies that object center detection can benefit from both pixel category contexts and pixel motion contexts, and most of the gain of global motion prediction in our approach comes from local motion supervision.

Ablation study: local motion prediction module. In HyMo, we use the LocalMotion model as the local motion prediction module by default. Due to the versatility of HyMo framework, the local motion prediction module can be replaced with other

TABLE V: Comparison of HyMo using LocalMotion and MotionNet as local motion prediction module.

Method	Local Motion Prediction	Local FDE	Global FDE
LocalMotion	LocalMotion	0.41 m	–
HyMo	LocalMotion	0.38 m	0.53 m
MotionNet	MotionNet	0.38 m	–
HyMo	MotionNet	0.37 m	0.53 m

TABLE VI: Ablation study on two of local motion modelings: homogeneous local motion and inhomogeneous local motion. Global motion prediction benefit more from homogeneous local motion contexts. LM FDE refers to local motion FDE, GM FDE refers to global motion FDE.

Method	Local Motion Type	LM FDE (m)	GM FDE (m)
PixelBaseline	homogeneous	0.41	–
	inhomogeneous	0.42	–
HyMo	homogeneous	0.38	0.53
	inhomogeneous	0.38	0.62

models like MotionNet [72]. Table V shows the results of HyMo using LocalMotion model and MotionNet model as the local motion prediction module. In both cases, HyMo is able to improve the local motion prediction performance. In terms of global motion prediction, the two models have the same performance.

Ablation study: homogeneous vs. inhomogeneous local motion. As discussed in Sec. III-B, we try two types of local motion: homogeneous local motion and inhomogeneous local motion. Homogeneous local motion regards the motion of points belonging to the same object to be the same. Homogeneous local motion is always the same as the global motion of the object. Inhomogeneous local motion is a finer model of point motion. Rotation of objects is considered, so for inhomogeneous local motion, motion of points belonging to the same object can be different.

Table VI shows the comparison between homogeneous and inhomogeneous local motion on local motion prediction and global motion prediction. Here LM FDE refers to the local motion FDE for vehicle pixels, GM FDE refers to the global motion FDE for vehicles. For local motion prediction, homogeneous and inhomogeneous local motion performs similarly. For global motion prediction, homogeneous local motion performs better than inhomogeneous local motion (0.53 m vs. 0.62 m FDE).

This result implies that, despite that inhomogeneous local motion is a better approximation of real local motion, inhomogeneous local motion is not necessarily a better context for global motion prediction. Instead, since the inhomogeneous local motion can be different from object global motion, the inhomogeneous local motion context can be inaccurate for global motion prediction. On the contrary, homogeneous local motion is always the same as object global motion, which can serve as a better context for global motion prediction.

Ablation study: joint-training vs. pre-training. We investigate the reason for the improvement from local motion in HyMo. As illustrated in Table VII, we compare two ways of using local motion: (1) we first train a local motion prediction model, then we train a global motion prediction model with backbone weights initialized by the weights in the trained local motion model. (2) jointly train local and global motion prediction. Pre-training and joint-training have similar object center detection AP, which means that local motion training helps backbone to extract more useful features is the main reason of improvement brought by local motion for object center detection. For global motion prediction, joint-training still outperforms pre-training. The reason might be: local motion supervision serves as a pixel-wise auxiliary supervision for global motion prediction.

TABLE VII: Global-local joint-training vs. local motion pre-training. We study different ways to use local motion information to help global motion prediction. We compare joint-training of global motion and local motion (HyMo) with using local-motion pretrained weights as network initialization. Joint-training and pre-training have similar object center detection performance, while joint-training performs better on motion prediction.

Method	Usage of Local Motion	AP^C_{2m}	FDE (m)
GlobalMotion	None	5.6	1.09
HyMo	pre-training	45.0	0.74
	joint-training	45.8	0.53

TABLE VIII: Ablation study on matching cost. Reg. cost + Cls. cost performs overall the best. Adding Pred. cost in matching interferes the training and results in bad object center detection and global motion prediction quality.

Reg. cost	Cls. cost	Pred. cost	AP^C_{2m}	AR^C_{2m}	FDE (m)
✓			41.2	77.7	0.71
✓	✓		45.8	60.4	0.53
✓	✓	✓	2.9	33.3	1.19

Ablation study: matching cost. For label assignment in global motion representation learning, we adopt bipartite matching. We study the three formations of the matching cost: (a) regression cost only, (b) regression + classification cost, (c) repression + classification + prediction cost.

Table VIII shows the comparison of the three formations. In terms of object center detection, both (a) and (b) perform satisfactorily. (a) has higher recall: 77.7 vs. 60.4AR while (b) has higher precision: 45.8 vs. 41.2AP. Detection performance of (c) is much worse than (a) and (b). In terms of global motion prediction, (b) performs the best and achieves 0.53 m FDE. (c) performs the worst.

The model with prediction cost performs poorly, we guess that there are two reasons. Firstly, global motion prediction relies on object center detection. When object center detection is not well-optimized, adding prediction cost to matching cost might introduce noise. Secondly, global motion is not a unique identifier of objects like center position. Multiple objects can have the same global motion (e.g., static), while different objects must have different center points.

Comparison between (a) and (b) shows that introducing classification cost improves AP while harms AR, which im-

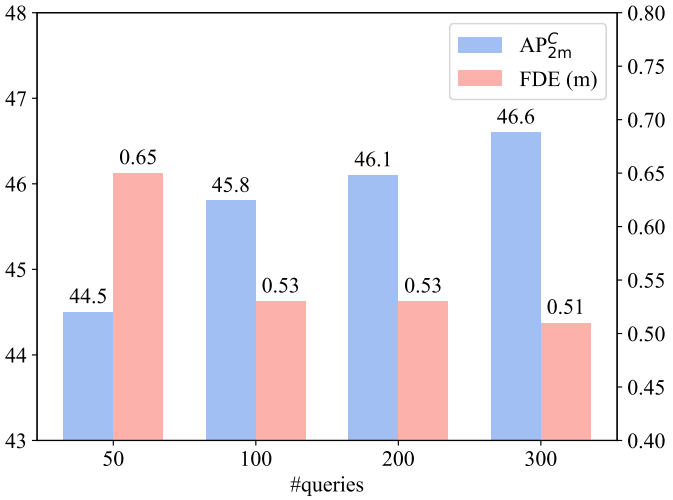

Fig. 7: Ablation study on the number of queries. Object center detection AP is improved with more queries. Global motion prediction with 50 queries is worse than 100, 200, and 300 queries.

plies that classification cost can help the ranking of detections.

Ablation study: number of decoder queries. As shown in Figure 7, we study how the number of queries affects object center detection and global motion prediction. Model with 50 queries performs worse than model with 100, 200, and 300 queries on both object center detection and global motion. For object center detection, AP is improved consistently with more queries. For global motion prediction, model with 100, 200, and 300 queries performs similarly.

F. Visualizations

We visualize HyMo predictions compared to ground-truths in Figure 6. For motion prediction, the arrows represent the displacement from the start point to the endpoint. Local motion predictions are generally better than global motion predictions. In Scene 3 and Scene 5, global motions of some objects are missed, while the local motion prediction successfully captures their motions. On the other hand, local motion predictions have more false positives than global motion predictions. This implies that the combination of local and global motion can be more robust. Further, in Scene 1 and Scene 5, we observe that HyMo detects moving objects that are missed in ground-truth annotations.

V. Limitations and Future Work

Our approach only utilizes LiDAR sensor data as input, without exploiting high-definition maps as well as data from other sensors (e.g., cameras). Thus, in real-world applications, our approach might perform unstably under adverse conditions, such as rain, snow, and heavy fog. In the future, we will study how to exploit multi-modality sensor data and high-definition maps for more robust prediction.

In HyMo model, we use a fully-convolutional backbone, and thus the temporal relations are not fully explored. We will study how to better explore temporal representations like adopting LSTM models in the backbone [24], [80]

Further, HyMo might also benefit from semi-supervised learning methods for dense prediction tasks [10], [46] since the local motion prediction task might suffer from the curse of dimensionality, and lack of high-quality labels.

VI. Conclusion

We present a hybrid motion representation learning framework (HyMo) that jointly performs global and local motion prediction. We demonstrate that local motion representations can improve global motion prediction by providing local contexts that are not explicitly exploited. Correspondingly, global motion representations can provide object contexts that are not encoded in local motion representations. Global motion prediction and local motion prediction can mutually benefit from each other. We build a motion prediction model that jointly learns global and local motion representations. Experimental results on the nuScenes dataset validate the effectiveness of the proposed framework.

References

[1] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social LSTM: human trajectory prediction in crowded spaces. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[2] Mayank Bansal, Alex Krizhevsky, and Abhijit S. Ogale. Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. In *Robotics: Science and Systems XV, University of Freiburg, Freiburg im Breisgau, Germany, June 22-26, 2019*, 2019.

[3] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[4] Zhaowei Cai and Nuno Vasconcelos. Cascade R-CNN: delving into high quality object detection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Proc. European Conference on Computer Vision*, 2020.

[6] Sergio Casas, Cole Gulino, Simon Suo, Katie Luo, Renjie Liao, and Raquel Urtasun. Implicit latent variable model for scene-consistent motion forecasting. In *Proc. European Conference on Computer Vision*, 2020.

[7] Sergio Casas, Wenjie Luo, and Raquel Urtasun. Intentnet: Learning to predict intention from raw sensor data. In *Proc. Conference on Robot Learning*, 2018.

[8] Sergio Casas, Abbas Sadat, and Raquel Urtasun. MP3: A unified model to map, perceive, predict and plan. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[9] Yuning Chai, Benjamin Sapp, Mayank Bansal, and Dragomir Anguelov. Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. In *Proc. Conference on Robot Learning*, 2019.

[10] Xiaokang Chen, Yuhui Yuan, Gang Zeng, and Jingdong Wang. Semi-supervised semantic segmentation with cross pseudo supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2613–2622, 2021.

[11] Hang Chu, Daiqing Li, David Acuna, Amlan Kar, Maria Shugrina, Xinkai Wei, Ming-Yu Liu, Antonio Torralba, and Sanja Fidler. Neural turtle graphics for modeling city road layouts. In *Proc. IEEE International Conference on Computer Vision*, 2019.

[12] Henggang Cui, Vladan Radosavljevic, Fang-Chieh Chou, Tsung-Han Lin, Thi Nguyen, Tzu-Kuo Huang, Jeff Schneider, and Nemanja Djuric. Multimodal trajectory predictions for autonomous driving using deep convolutional networks. In *Proc. IEEE International Conference on Robotics and Automation*, 2019.

[13] Zhigang Dai, Bolun Cai, Yugeng Lin, and Junying Chen. UP-DETR: unsupervised pre-training for object detection with transformers. *arXiv*, abs/2011.09094, 2020.

[14] Jiajun Deng, Shaoshuai Shi, Peiwei Li, Wengang Zhou, Yanyong Zhang, and Houqiang Li. Voxel R-CNN: towards high performance voxel-based 3d object detection. In *Proc. AAAI Conference on Artificial Intelligence*, 2021.

[15] Jiajun Deng, Wengang Zhou, Yanyong Zhang, and Houqiang Li. From multi-view to hollow-3d: Hallucinated hollow-3d R-CNN for 3d object detection. *IEEE Trans. Circuits Syst. Video Technol.*, 2021.

[16] Nemanja Djuric, Vladan Radosavljevic, Henggang Cui, Thi Nguyen, Fang-Chieh Chou, Tsung-Han Lin, and Jeff Schneider. Motion prediction of traffic actors for autonomous driving using deep convolutional networks. *arXiv*, abs/1808.05819, 2018.

[17] 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 *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[18] Peng Gao, Minghang Zheng, Xiaogang Wang, Jifeng Dai, and Hongsheng Li. Fast convergence of DETR with spatially modulated co-attention. *arXiv*, abs/2101.07448, 2021.

[19] Thomas Gilles, Stefano Sabatini, Dzmitry Tsishkou, Bogdan Stanciulescu, and Fabien Moutarde. Home: Heatmap output for future motion estimation. 2021.

[20] Ross B. Girshick. Fast R-CNN. In *Proc. IEEE International Conference on Computer Vision*, 2015.

[21] Junru Gu, Chen Sun, and Hang Zhao. Densetnt: End-to-end trajectory prediction from dense goal sets. In *Proc. IEEE International Conference on Computer Vision*, pages 15303–15312, 2021.

[22] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social GAN: socially acceptable trajectories with generative adversarial networks. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[23] Hao He, Hengchen Dai, and Naiyan Wang. Ust: Unifying spatio-temporal context for trajectory prediction in autonomous driving. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5962–5969. IEEE, 2020.

[24] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[25] Namdar Homayounfar, Justin Liang, Wei-Chiu Ma, Jack Fan, Xinyu Wu, and Raquel Urtasun. Dagmapper: Learning to map by discovering lane topology. In *Proc. IEEE International Conference on Computer Vision*, 2019.

[26] Namdar Homayounfar, Wei-Chiu Ma, Shrinidhi Kowshika Lakshmikanth, and Raquel Urtasun. Hierarchical recurrent attention networks for structured online maps. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[27] Joey Hong, Benjamin Sapp, and James Philbin. Rules of the road: Predicting driving behavior with a convolutional model of semantic interactions. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[28] Lichao Huang, Yi Yang, Yafeng Deng, and Yinan Yu. Densebox: Unifying landmark localization with end to end object detection. *arXiv*, abs/1509.04874, 2015.

[29] Shan Jiayao, Sifan Zhou, Yubo Cui, and Zheng Fang. Real-time 3d single object tracking with transformer. *IEEE Transactions on Multimedia*, 2022.

[30] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. International Conference on Learning Representations*, 2015.

[31] Tao Kong, Fuchun Sun, Huaping Liu, Yuning Jiang, and Jianbo Shi. Foveabox: Beyond anchor-based object detector. *arXiv*, abs/1904.03797, 2019.

[32] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L. Waslander. Joint 3d proposal generation and object detection from view aggregation. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018.

[33] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *Proc. European Conference on Computer Vision*, 2018.

[34] Hei Law, Yun Teng, Olga Russakovsky, and Jia Deng. Cornernet-lite: Efficient keypoint based object detection. In *BMVC*. BMVA Press, 2020.

[35] Jianan Li, Xiaodan Liang, Jianshu Li, Yunchao Wei, Tingfa Xu, Jiashi Feng, and Shuicheng Yan. Multistage object detection with group recursive learning. *IEEE Transactions on Multimedia*, 2018.

[36] Lingyun Luke Li, Bin Yang, Ming Liang, Wenyuan Zeng, Mengye Ren, Sean Segal, and Raquel Urtasun. End-to-end contextual perception and prediction with interaction transformer. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020.

[37] Yanghao Li, Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. Scale-aware trident networks for object detection. In *Proc. IEEE International Conference on Computer Vision*, pages 6054–6063, 2019.

[38] Justin Liang, Namdar Homayounfar, Wei-Chiu Ma, Shenlong Wang, and Raquel Urtasun. Convolutional recurrent network for road boundary extraction. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[39] Ming Liang, Bin Yang, Rui Hu, Yun Chen, Renjie Liao, Song Feng, and Raquel Urtasun. Learning lane graph representations for motion forecasting. In *Proc. European Conference on Computer Vision*, 2020.

[40] Ming Liang, Bin Yang, Wenyuan Zeng, Yun Chen, Rui Hu, Sergio Casas, and Raquel Urtasun. Pnpnet: End-to-end perception and prediction with tracking in the loop. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[41] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *TPAMI*, 2020.

[42] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: single shot multibox detector. In *Proc. European Conference on Computer Vision*, 2016.

[43] Yingfei Liu, Tiancai Wang, Xiangyu Zhang, and Jian Sun. Petr: Position embedding transformation for multi-view 3d object detection. *arXiv*, 2022.

[44] Ze Liu, Zheng Zhang, Yue Cao, Han Hu, and Xin Tong. Group-free 3d object detection via transformers. *Proc. IEEE International Conference on Computer Vision*, 2021.

[45] Xin Lu, Buyu Li, Yuxin Yue, Quanquan Li, and Junjie Yan. Grid R-CNN. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[46] Minnan Luo, Xiaojun Chang, Liqiang Nie, Yi Yang, Alexander G Hauptmann, and Qinghua Zheng. An adaptive semisupervised feature analysis for video semantic recognition. *IEEE transactions on cybernetics*, 48(2):648–660, 2017.

[47] Wenjie Luo, Bin Yang, and Raquel Urtasun. Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[48] Karttikeya Mangalam, Harshayu Girase, Shreyas Agarwal, Kuan-Hui Lee, Ehsan Adeli, Jitendra Malik, and Adrien Gaidon. It is not the journey but the destination: Endpoint conditioned trajectory prediction. In *Proc. European Conference on Computer Vision*, pages 759–776. Springer, 2020.

[49] Gellért Máttyus, Shenlong Wang, Sanja Fidler, and Raquel Urtasun. Enhancing road maps by parsing aerial images around the world. In *Proc. IEEE International Conference on Computer Vision*, 2015.

[50] Depu Meng, Xiaokang Chen, Zejia Fan, Gang Zeng, Houqiang Li, Yuhui Yuan, Lei Sun, and Jingdong Wang. Conditional DETR for fast training convergence. In *Proc. IEEE International Conference on Computer Vision*, 2021.

[51] Gregory P. Meyer, Ankit Laddha, Eric Kee, Carlos Vallespi-Gonzalez, and Carl K. Wellington. Lasernet: An efficient probabilistic 3d object detector for autonomous driving. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[52] Ishan Misra, Rohit Girdhar, and Armand Joulin. An end-to-end transformer model for 3d object detection. *Proc. IEEE International Conference on Computer Vision*, 2021.

[53] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. 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 Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Proc. Advances in Neural Information Processing Systems*, 2019.

[54] Tung Phan-Minh, Elena Corina Grigore, Freddy A Boulton, Oscar Beijbom, and Eric M Wolff. Covernet: Multimodal behavior prediction using trajectory sets. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 14074–14083, 2020.

[55] John Phillips, Julieta Martinez, Ioan Andrei Barsan, Sergio Casas, Abbas Sadat, and Raquel Urtasun. Deep multi-task learning for joint localization, perception, and prediction. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[56] Charles R. Qi, Or Litany, Kaiming He, and Leonidas J. Guibas. Deep hough voting for 3d object detection in point clouds. In *Proc. IEEE International Conference on Computer Vision*, 2019.

[57] Yu Qiu, Yun Liu, Yanan Chen, Jianwen Zhang, Jinchao Zhu, and Jing Xu. A2sppnet: Attentive atrous spatial pyramid pooling network for salient object detection. *IEEE Transactions on Multimedia*, 2022.

[58] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[59] Joseph Redmon and Ali Farhadi. YOLO9000: better, faster, stronger. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[60] Eike Rehder and Horst Kloeden. Goal-directed pedestrian prediction. In *Proc. IEEE International Conference on Computer Vision Workshop*, pages 50–58, 2015.

[61] Eike Rehder, Florian Wirth, Martin Lauer, and Christoph Stiller. Pedestrian prediction by planning using deep neural networks. In *Proc. IEEE International Conference on Robotics and Automation*, pages 5903–5908. IEEE, 2018.

[62] Nicholas Rhinehart, Rowan McAllister, Kris Kitani, and Sergey Levine. Precog: Prediction conditioned on goals in visual multi-agent settings. In *Proc. IEEE International Conference on Computer Vision*, pages 2821–2830, 2019.

[63] Alexandre Robicquet, Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. Learning social etiquette: Human trajectory understanding in crowded scenes. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Proc. European Conference on Computer Vision*, 2016.

[64] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. PV-RCNN: point-voxel feature set abstraction for 3d object detection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[65] Shaoshuai Shi, Li Jiang, Jiajun Deng, Zhe Wang, Chaoxu Guo, Jianping Shi, Xiaogang Wang, and Hongsheng Li. PV-RCNN++: point-voxel feature set abstraction with local vector representation for 3d object detection. *arXiv*, 2021.

[66] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[67] Chen Sun, Per Karlsson, Jiajun Wu, Joshua B. Tenenbaum, and Kevin Murphy. Stochastic prediction of multi-agent interactions from partial observations. In *Proc. International Conference on Learning Representations*, 2019.

[68] Chen Sun, Abhinav Shrivastava, Carl Vondrick, Rahul Sukthankar, Kevin Murphy, and Cordelia Schmid. Relational action forecasting. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[69] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. FCOS: fully convolutional one-stage object detection. In *Proc. IEEE International Conference on Computer Vision*, 2019.

[70] 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.

[71] Yue Wang, Vitor Campagnolo Guizilini, Tianyuan Zhang, Yilun Wang, Hang Zhao, and Justin Solomon. Detr3d: 3d object detection from multi-view images via 3d-to-2d queries. In *Proc. Conference on Robot Learning*, pages 180–191. PMLR, 2022.

[72] Pengxiang Wu, Siheng Chen, and Dimitris N. Metaxas. Motionnet: Joint perception and motion prediction for autonomous driving based on bird's eye view maps. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[73] Bin Yang, Ming Liang, and Raquel Urtasun. HDNET: exploiting HD maps for 3d object detection. In *Proc. Conference on Robot Learning*, 2018.

[74] Bin Yang, Wenjie Luo, and Raquel Urtasun. PIXOR: real-time 3d object detection from point clouds. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[75] Jinrong Yang, Songtao Liu, Zeming Li, Xiaoping Li, and Jian Sun. Real-time object detection for streaming perception. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 5385–5395, 2022.

[76] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. STD: sparse-to-dense 3d object detector for point cloud. In *Proc. IEEE International Conference on Computer Vision*, 2019.

[77] Maosheng Ye, Tongyi Cao, and Qifeng Chen. TPCN: temporal point cloud networks for motion forecasting. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[78] Jiahui Yu, Yuning Jiang, Zhangyang Wang, Zhimin Cao, and Thomas S. Huang. Unitbox: An advanced object detection network. In *MM*, 2016.

[79] Wenyuan Zeng, Ming Liang, Renjie Liao, and Raquel Urtasun. Lanercnn: Distributed representations for graph-centric motion forecasting. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2021.

[80] Dalin Zhang, Lina Yao, Kaixuan Chen, Sen Wang, Xiaojun Chang, and Yunhao Liu. Making sense of spatio-temporal preserving representations for eeg-based human intention recognition. *IEEE transactions on cybernetics*, 50(7):3033–3044, 2019.

[81] Gongjie Zhang, Zhipeng Luo, Yingchen Yu, Kaiwen Cui, and Shijian Lu. Accelerating DETR convergence via semantic-aligned matching. *arXiv*, abs/2203.06883, 2022.

[82] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z. Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[83] 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 *Proc. Conference on Robot Learning*, volume 155, pages 895–904, 2020.

[84] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv*, abs/1904.07850, 2019.

[85] Benjin Zhu, Zhengkai Jiang, Xiangxin Zhou, Zeming Li, and Gang Yu. Class-balanced grouping and sampling for point cloud 3d object detection. *arXiv*, 2019.

[86] Chenchen Zhu, Fangyi Chen, Zhiqiang Shen, and Marios Savvides. Soft anchor-point object detection. In *Proc. European Conference on Computer Vision*, 2020.

[87] Chenchen Zhu, Yihui He, and Marios Savvides. Feature selective anchor-free module for single-shot object detection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[88] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: deformable transformers for end-to-end object detection. *arXiv*, abs/2010.04159, 2020.

[89] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: deformable transformers for end-to-end object detection. In *Proc. International Conference on Learning Representations*, 2021.