MultiXNet: Multiclass Multistage Multimodal Motion Prediction

Nemanja Djuric, Henggang Cui, Zhaoen Su, Shangxuan Wu, Huahua Wang,
Fang-Chieh Chou, Luisa San Martin, Song Feng, Rui Hu, Yang Xu, Alyssa Dayan,
Sidney Zhang, Brian C. Becker, Gregory P. Meyer, Carlos Vallespi-Gonzalez, Carl K. Wellington
Uber Advanced Technologies Group
{ndjuric, hcui2, suzhaoen, shangxuan.wu, anteaglewang, fchou, luisasm}@uber.com
{songf, rui.hu, yang.xu, ada, sidney, bbecker, gmeyer, cvallespi, cwellington}@uber.com

***Abstract*— One of the critical pieces of the self-driving puzzle is understanding the surroundings of a self-driving vehicle (SDV) and predicting how these surroundings will change in the near future. To address this task we propose MultiXNet, an end-to-end approach for detection and motion prediction based directly on lidar sensor data. This approach builds on prior work by handling multiple classes of traffic actors, adding a jointly trained second-stage trajectory refinement step, and producing a multimodal probability distribution over future actor motion that includes both multiple discrete traffic behaviors and calibrated continuous position uncertainties. The method was evaluated on large-scale, real-world data collected by a fleet of SDVs in several cities, with the results indicating that it outperforms existing state-of-the-art approaches.**

I. INTRODUCTION

Predicting future states of other traffic actors, such as vehicles, pedestrians, and bicyclists, represents a key capability for the self-driving technology. This is a challenging task, found to play an important role in accident avoidance both for human drivers [1], [2] and for their autonomous counterparts [3]. Within the context of a self-driving vehicle (SDV) it is important to capture the range of possibilities for other actors, and not just a single most likely trajectory. Consider an opposing vehicle approaching an intersection, which may continue driving straight or turn in front of the SDV (exemplified in Fig. 1). To ensure safety, the SDV needs to accurately reason about both of these possible modes and modulate its behavior accordingly. In addition to the discrete modes, a downstream motion planner may react differently to a prediction depending on the position uncertainty within a predicted trajectory. As an example, if an opposing vehicle looks like it might take a wide turn and come into the SDV's lane, the SDV can preemptively slow down to reduce the risk. On the other hand, if the prediction shows confidence that the opposing vehicle will stay in its lane the SDV could choose to maintain its current speed.

Bringing the above requirements together, Fig. 1 shows an example of the task addressed by this work. The input is a map and a sequence of lidar data which are projected into a common global coordinate frame using the SDV pose. The output is a multimodal distribution over potential future states for the other actors in the scene. An important challenge is that various actor types such as pedestrians and vehicles exhibit significantly different behavior, while a deployed approach needs to handle all actors present in a given scene.

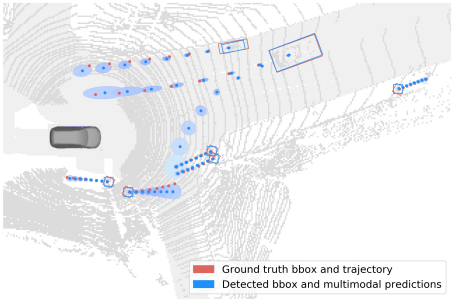

Fig. 1: Example output of the proposed MultiXNet model, showing detections and multimodal, uncertainty-aware motion predictions for multiple actor types overlaid on top of lidar and map data, including pedestrians on the sidewalks and a vehicle and a bicyclist approaching the SDV; note that the turning vehicle prediction is a low-probability trajectory that is unlikely, yet still possible to observe in the real world

Prior work in this area has demonstrated strong performance by using end-to-end methods that jointly learn detection and motion prediction directly from sensor data [4], [5], including the addition of a jointly learned refinement stage of the network that leads to improved trajectory prediction [6]. However, these approaches have generally focused only on vehicles and produce a single trajectory rather than full motion distributions. More recent work has shown the ability to learn a continuous distribution directly from sensor data for multiple classes [7], but the distributions are not multimodal. Prediction methods that operate on detections rather than the raw sensor data have shown improved performance by introducing multiclass predictions [8], estimates of uncertainty [9], [10], or incorporating multiple modes [11]. While each of these concepts has been considered individually, this work looks to unify them into a single approach which we empirically show to outperform the competing baselines.

Our work builds on IntentNet [5] to produce an end-to-end motion prediction model with the following contributions:

- joint detection and motion prediction of multiple actor classes: vehicles, pedestrians, and bicyclists;
- multimodal trajectory prediction to capture distinct potential future trajectories of traffic actors along with their corresponding continuous position uncertainties.

Using a large-scale, real-world data set, the proposed approach was shown to outperform the current state-of-the-art.

II. RELATED WORK

Object detection is a critical task for an SDV system, with a number of papers proposed recently in the literature. A popular approach is using a bird's-eye view (BEV) representation, where lidar points are encoded in 3D voxels [5], which has a strong benefit of being a range-invariant representation of objects. PointPillars [12] proposed to learn the BEV encoding through a computation scheme that provides better speed while keeping accuracy high. Range view (RV) is another popular lidar point representation that provides a compact input while preserving all sensor information. The authors of [13] showed that RV is good at detection of both near- and long-range objects, which can further be improved by combining a camera image with RV lidar [14]. Recent work applies both BEV and RV representations [15], [16], extracting features using separate branches of the network that are fused at a later stage. This fusion method preserves information for both near- and long-range objects, at the cost of a more complex and heavy network structure. In this work we focus on the BEV approach and discuss several ideas on how to improve the current state-of-the-art.

Movement prediction is another major topic in the SDV community. Typically, the prediction models take current and past detections as inputs, and then output trajectories for the next several seconds. A common approach is to train recurrent models to process the inputs and extract learned features [17], [18], [19], [20], [21], [22]. A number of methods have been proposed that take actor surroundings and other contextual information through BEV images as an input, and extract useful scene features using convolutional neural nets (CNNs) [4], [8], [9], [10], [11], [18]. Interestingly, the majority of former research on trajectory prediction has focused on predicting the motion of a particular type of road actor (e.g., vehicle or pedestrian). However, multiple types of traffic actors exist together on public roads, and SDVs need to accurately predict all relevant actors' motions in order to drive safely. Moreover, different actor types have distinct motion patterns (e.g., bicyclists and pedestrians behave quite differently [8]), and it is important to model them separately. A few recent papers tackled this challenge using recurrent methods [18], [23], [24]. However, unlike in our work, an existence of a detection system was assumed and they were not trained end-to-end using raw sensor data.

Another aspect of the prediction task that is important for ensuring safe SDV operations is modeling the stochasticity of traffic behavior, either by considering multimodality of actor movement (e.g., whether they are going to turn left or right at an intersection) or position uncertainty within a single mode. When it comes to the multimodality of future trajectories there are two common classes of approaches. The first is the use of generative models, either explicitly with conditional variational autoencoders [17], [18], [25] or implicitly with generative adversarial networks [19], [20], [21], [22], [26]. Once trained, trajectories are predicted by sampling from the learned distribution at inference time. The generative models often require the system to draw many samples to ensure good coverage in the trajectory space (e.g., as many as 20 in [19], [21]), which may be impractical for time-critical applications. The second category of approaches directly predicts a fixed number of trajectories along with their probabilities in a single-shot manner [9], [11], [27], [28]. The trajectories and probabilities are jointly trained with a combination of regression and classification losses, and are much more efficient than the alternatives. As a result, most applied work follows the one-shot approach [9], [27].

In addition to multimodality, it is important to capture the uncertainty of actor motion within a trajectory mode. This can be achieved by explicitly modeling each trajectory as a probability distribution, for example by modeling trajectory waypoints using Gaussians [9], [10], [18], [29]. Following a different paradigm, some researchers have proposed non-parametric approaches [30] to directly predict an occupancy map. While parametric approaches can easily be cast into cell occupancy space the reverse is not necessarily true, limiting the applicability of such output representations in downstream modules of the SDV system.

Instead of using independent detection and motion forecasting models, some recent work has proposed to train them jointly in an end-to-end fashion, taking raw sensor data as inputs. This approach was pioneered in the FaF model [4], while IntentNet [5] further included map data as an input and proposed to predict both actor trajectories and their high-level intents. The authors of [31] further extended this idea to an end-to-end model that also includes motion planning. SpAGNN [6] introduced a two-stage model with Rotated Region of Interest (RROI) cropping, a graph neural network module to encode actor relations, as well as modeling the uncertainty of future trajectories. MotionNet [32] used a spatial-temporal pyramid network to jointly perform detection and motion prediction for each BEV grid cell. LaserFlow [7] proposed an end-to-end model using multi-frame RV lidar inputs, unlike the other methods using BEV representations, which can also perform prediction on multiple actor types. Compared to our method, most of the above end-to-end methods do not consider motion prediction on diverse road actor types, and none of them addresses the multimodal nature of possible future trajectories. The earlier work has clearly shown the promise of end-to-end approaches, with researchers looking at various aspects to improve the prediction performance. In this paper we propose the first model to bring these key ideas together, and show in the experimental section the benefits over the baselines.

III. PROPOSED APPROACH

In this section we describe our end-to-end method for joint detection and prediction, called MultiXNet. We first describe the existing state-of-the-art IntentNet architecture [5], followed by a discussion of our proposed improvements.

A. Baseline end-to-end detection and prediction

1) Input representation: In Fig. 2 we show the lidar and map inputs to the network. We assume a lidar sensor installed on the SDV that provides measurements at regular

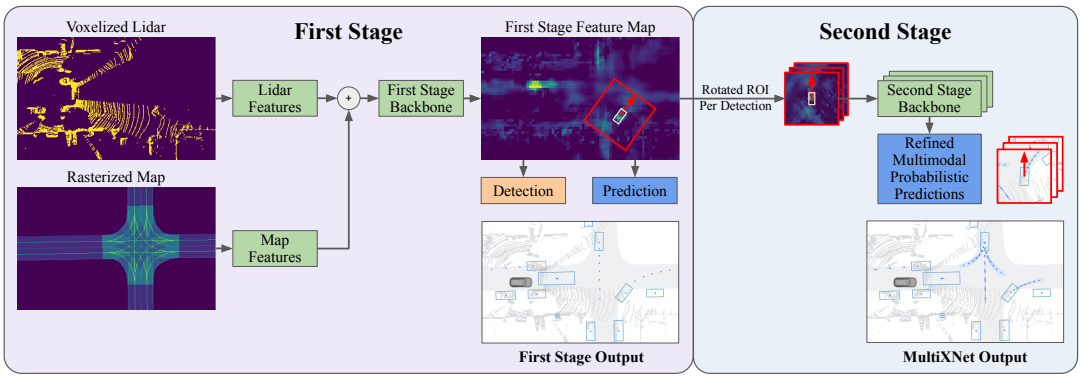

Fig. 2: Overview of the MultiXNet architecture, where the first-stage network corresponding to IntentNet [5] outputs actor detections and their unimodal motion prediction, while the second stage refines predictions to be multimodal and uncertainty-aware; note that the first-stage prediction of the right-turning vehicle is incorrect, and the second stage improves its prediction

time intervals. At time t a lidar sweep $\mathcal{S}_t$ comprises a set of 3D lidar returns represented by their (x, y, z) locations. Following [5], we encode the lidar data $\mathcal{S}_t$ in a BEV image centered on the SDV, with voxel sizes of Δ_L and Δ_W along the forward x- and left y-axes, respectively, and Δ_V along the vertical axis (representing the image channels). Each voxel encodes binary information about whether or not there exists at least one lidar return inside that voxel. In addition, to capture temporal information we encode the $T-1$ past lidar sweeps $\{\mathcal{S}_{t-T+1}, \ldots, \mathcal{S}_{t-1}\}$ into the same BEV frame using their known SDV poses, and stack them together along the channel (or vertical) dimension. Assuming we consider an area of length L, width W, and height V, this yields an input image of size $\left\lceil \frac{L}{\Delta_L} \right\rceil \times \left\lceil \frac{W}{\Delta_W} \right\rceil \times T \left\lceil \frac{V}{\Delta_V} \right\rceil$.

Moreover, let us assume we have access to a high-definition map of the operating area around the SDV denoted by $\mathcal{M}$. As shown in Fig. 2, we encode static map elements from $\mathcal{M}$ in the same BEV frame as introduced above. These include driving paths, crosswalks, lane and road boundaries, intersections, driveways, and parking lots, where each element is encoded as a binary mask in its own separate channel. This results in a total of seven map additional channels, which are processed by a few convolutional layers before being stacked with the processed lidar channels, to be used as a BEV input to the rest of the network, as described in [5].

2) Network architecture and output: The input BEV image can be viewed as a top-down grid representation of the SDV's surroundings, with each grid cell comprising input features encoded along the channel dimensions. As in [5], this image is then processed by a sequence of 2-D convolutional layers to produce a final layer that contains learned features for each cell location. Following an additional 1×1 convolutional layer, for each cell we predict two sets of outputs, representing object detection and its movement prediction (in the following text we denote a predicted value by the hat-notation $\hat{*}$). In particular, the detection output for a cell centered at (x, y) comprises an existence probability $\hat{p}$, oriented bounding box represented by its center $\hat{\mathbf{c}}_0 = (\hat{c}_{x0}, \hat{c}_{y0})$ relative to the center of the grid cell, size represented by length $\hat{l}$ and width $\hat{w}$, and heading $\hat{\theta}_0$ relative to the x-axis, parameterized as a tuple $(\sin\hat{\theta}_0, \cos\hat{\theta}_0)$. In addition, the prediction output is composed of bounding box centers (or *waypoints*) $\hat{\mathbf{c}}_h = (\hat{c}_{xh}, \hat{c}_{yh})$ and headings $\hat{\theta}_h$ at H future time horizons, with $h \in \{1, \ldots, H\}$. A full set of H waypoints is denoted as a trajectory $\hat{\boldsymbol{\tau}} = \{\hat{\mathbf{c}}_h, \hat{\theta}_h\}_{h=1}^{H}$, where the bounding box size is considered constant across the entire prediction horizon.

3) Loss: As discussed in [5], the loss at a certain time step consists of detection and prediction losses computed over all BEV cells. When it comes to the per-pixel detection loss, a binary focal loss $\ell_{focal}(\hat{p}) = (1-\hat{p})^{\gamma} \log \hat{p}$ is used for the probability of a ground-truth object [33], where we empirically found good performance with hyper-parameter γ set to 2. Moreover, when there exists a ground-truth object in a particular cell a smooth-ℓ_1 regression loss $\ell_1(\hat{v} - v)$ is used for all bounding box parameters (i.e., center, size, and heading), where the loss is computed between the predicted value $\hat{v}$ and the corresponding ground truth v. The smooth-ℓ_1 regression loss is also used to capture prediction errors of future bounding box centers and headings. We refer to a cell containing an object as a *foreground* (fg) cell, and a *background* (bg) cell otherwise. Then, the overall loss at horizon h for a foreground cell $\mathcal{L}_{fg(h)}$ is computed as

$$\begin{aligned}\mathcal{L}_{fg(h)} = \; & 1_{h=0}\Big(\ell_{focal}(\hat{p}) + \ell_1(\hat{l} - l) + \ell_1(\hat{w} - w)\Big) + \\ & \ell_1(\hat{c}_{xh} - c_{xh}) + \ell_1(\hat{c}_{yh} - c_{yh}) + \\ & \ell_1(sin\hat{\theta}_h - sin\theta_h) + \ell_1(cos\hat{\theta}_h - cos\theta_h),\end{aligned} \tag{1}$$

where 1_c equals 1 if the condition c holds and 0 otherwise. The loss for a background cell equals $\mathcal{L}_{bg} = \ell_{focal}(1 - \hat{p})$.

Lastly, to enforce a lower error tolerance for earlier horizons we multiply the per-horizon losses by fixed weights that are gradually decreasing for future timesteps, and the per-horizon losses are aggregated to obtain the final loss,

$$\mathcal{L} = 1_{\text{bg cell}}\mathcal{L}_{bg} + 1_{\text{fg cell}} \sum_{h=0}^{H} \lambda^h \mathcal{L}_{fg(h)}, \tag{2}$$

where $\lambda \in (0, 1)$ is a constant decay factor (set to 0.97 in our experiments). The loss contains both detection and prediction components, and all model parameters are learned jointly in an end-to-end manner.

B. Improving end-to-end motion prediction

In this section we present an end-to-end method that we build on the approach presented in the previous section, extending it to significantly improve its prediction performance.

1) Uncertainty-aware loss: In addition to predicting trajectories, an important task in autonomous driving is the estimation of their spatial uncertainty. This is useful for fusion of results from multiple predictors, and is also consumed by a motion planner to improve SDV safety. In earlier work [10] it was proposed as a fine-tuning step following training of a model that only considered trajectory waypoints without uncertainties. Then, by freezing the main prediction weights or setting a low learning rate, the uncertainty module was trained without hurting the overall prediction performance.

In this paper we describe a method that learns trajectories and uncertainties jointly, where we decompose the position uncertainty in the along-track (AT) and cross-track (CT) directions [34]. In particular, a predicted waypoint $\hat{\mathbf{c}}_h$ is projected along AT and CT directions by considering the ground-truth heading θ_h, and the errors along these directions are assumed to follow a Laplace distribution $Laplace(\mu, b)$, where mean μ and diversity b are the Laplace parameters. We assume that AT and CT errors are independent, with each having a separate set of Laplace parameters. Taking AT as an example and assuming an error value $\hat{e}_{AT}$, this defines a Laplace distribution $Laplace(\hat{e}_{AT}, \hat{b}_{AT})$. Then, we minimize the loss by minimizing the Kullback–Leibler (KL) divergence between the ground-truth $Laplace(0, b_{AT})$ and the predicted $Laplace(\hat{e}_{AT}, \hat{b}_{AT})$, computed as follows [35],

$$KL_{AT} = \log \frac{\hat{b}_{AT}}{b_{AT}} + \frac{b_{AT} \exp\left(-\frac{|\hat{e}_{AT}|}{b_{AT}}\right) + |\hat{e}_{AT}|}{\hat{b}_{AT}} - 1. \quad (3)$$

Similarly, KL_{CT} can be computed for the CT errors, and we then use KL_{AT} and KL_{CT} instead of the smooth-ℓ_1 loss for bounding box centers introduced in the previous section. In the experimental section we compare the choice of Laplace versus Gaussian distribution, as well as the choice of KL divergence loss versus negative log-likelihood loss. The results show the benefit of optimizing Laplace KL divergence to model prediction errors.

An important question is a choice of ground-truth diversities b_{AT} and b_{CT}. In earlier detection work [36] a percentage of label area covered by lidar points was used, however this may not be the best choice for the prediction task as the prediction difficulty and uncertainty are expected to grow with longer horizons. To account for this, we linearly increase the ground-truth diversity with time as

$$b_*(t) = \alpha_* + \beta_* t, \quad (4)$$

where parameters α_* and β_* are empirically determined, with separate parameters for AT and for CT components. This is achieved by training models with varying α_* and β_* parameters and choosing the parameter set for which the reliability diagrams [10] indicate that the model outputs are the most calibrated, discussed in Sec. IV-B.

2) Second-stage trajectory refinement: As shown in Fig. 2, following the detection and prediction inference described in Sec. III-A we perform further refinement of the motion predictions for the detected objects, where we take the detected objects and infer their improved future trajectories through further processing. The refinement network, which we refer to as the *second stage* of the model, discards the first-stage trajectory predictions and takes the inferred object center $\hat{\mathbf{c}}_0$ and heading $\hat{\theta}_0$, as well as the final feature layer from the main network. Then, it crops and rotates learned features for each actor, such that the actor is oriented pointing up in the rotated image [8], [10], [37] as illustrated in Fig. 2. The RROI feature map is then fed through a lightweight CNN network before the final prediction of future trajectory and uncertainty is performed. Both first- and second-stage networks are trained jointly, using the full loss $\mathcal{L}$ in the first stage and only the future prediction loss in the second stage, where the second-stage predictions are used as the final output trajectories. The second stage is fully differentiable and therefore gradients flow through the second stage into the first stage.

The proposed method has several advantages. First, the output representation can be standardized in the actor frame. In the first-stage model the trajectories can radiate in any direction from the actor position, while in the actor frame the majority of the future trajectories grow from the origin forward. In addition, the second stage can concentrate on extracting features for a single actor of interest and discard irrelevant information. Lastly, note that the predictions from the first and second stage can both be used in the final system, depending on the latency requirements. For example, because the second-stage predictions incur additional latency cost, the system can use the first-stage predictions for some actors, and only run the second-stage for actors where this refinement is deemed important.

3) Multimodal trajectory prediction: Traffic behavior is inherently multimodal, as traffic actors at any point may make one of several movement decisions. Modeling such behavior is an important task in the self-driving field, with several interesting ideas being proposed in the literature [9], [11], [27], [28]. In this paper we address this problem, and describe an approach to output a fixed number of trajectories for each detected actor along with their probabilities. In particular, instead of outputting a single predicted trajectory in the second stage for each detected actor, the model outputs a fixed number of M trajectories. Let us denote trajectory modes output by the model as $\{\hat{\boldsymbol{\tau}}_m\}_{m=1}^M$ and their probabilities as $\{\hat{p}_m\}_{m=1}^M$. First, we identify one of the M modes as the *ground-truth mode* m_{gt}, for which purpose we designed a novel direction-based policy to decide the ground-truth mode. More specifically, we compute an angle $\Delta_\theta = \theta_H - \theta_0$ between the last and the current ground-truth heading, where $\Delta_\theta \in (-\pi, \pi]$. Then, we divide the range $(-\pi, \pi]$ into M bins and decide m_{gt} based on where Δ_θ falls. In this way, during training each mode is specialized to be responsible for a distinct behavior (e.g., for $M = 3$ we have left-turning, right-turning, and going-straight modes).

Note that there is an option to learn the modes through an Expectation-Maximization scheme, or by directly learning a mixture-density network as discussed in [11]. However, we found the proposed approach to perform best in practice.

Given the predictions and the ground-truth trajectory, and using a similar approach as discussed in [11], the multimodal trajectory loss consists of a trajectory loss of the m_{gt}-th trajectory mode as described in Sec. III-B.1 and a cross-entropy loss for the trajectory mode probabilities. Lastly, we continue to use unimodal prediction loss in the first stage to improve the model training, and the multimodal trajectory loss is only applied to train the second-stage network. Note that the accuracy of first-stage detections directly impacts the quality of second-stage predictions, as the second-stage is using the first-stage outputs as its inputs. However, we did not observe issues due to this dependency, and the experimental results show that the second-stage refinement leads to improved prediction performance.

4) Handling multiple actor types: Unlike earlier work [5] that mostly focused on a single traffic actor type, we model the behavior of multiple actor types simultaneously, focusing on vehicles, pedestrians, and bicyclists. This is done by separating three sets of outputs, one for each type, after the backbone network computes the shared BEV learned features shown in Fig. 2. Note that we found that the model achieves the best performance when each class has additional, separate processing following the learned features. Handling all actors using a single model and in a single pass simplifies the SDV system significantly, and helps ensure safe and effective operations. It is important to emphasize that in the case of pedestrians and bicyclists we found that a unimodal output results in the best performance, and we do not use the multimodal loss nor the refinement stage for these traffic actors. Thus, in our experiments we set $M = 3$ for vehicles and $M = 1$ for the other actor types. Then, the final loss of the model is a sum of per-type losses, with each per-type loss comprising the detection loss as described in Sec. III-A.3, as well as the uncertainty-aware trajectory loss described in Sec. III-B.1, Sec. III-B.2, and Sec. III-B.3.

IV. EXPERIMENTS

A. Experimental setup

Following earlier work [6] we evaluated the proposed approach using the open-sourced nuScenes data [38] and proprietary ATG4D data. The ATG4D data was collected by a fleet of SDVs across several cities in North America using a 64-beam, roof-mounted lidar sensor. It contains over 1 million frames collected from $5{,}500$ different scenarios, each scenario being a sequence of 250 frames captured at $10Hz$. The labels are precise tracks of 3D bounding boxes at a maximum range of 100 meters from the data-collecting vehicle. Vehicles are the most common actor type in the data set, with 3.2x fewer pedestrians and 15x fewer bicyclists.

We set the parameters of the BEV image to $L = 150m$, $W = 100m$, $V = 3.2m$, $\Delta_L = 0.16m$, $\Delta_W = 0.16m$, $\Delta_V = 0.2m$, and use $T = 10$ sweeps to predict $H = 30$ future states at $10Hz$ (resulting in predictions that are $3s$ long). For the second stage, we cropped a $40m \times 40m$ region around each actor. The models were implemented in PyTorch [39] and trained end-to-end with 16 GPUs, a per-GPU batch size of 2, Adam optimizer [40], and an initial learning rate of 2e-4, training for 2 epochs completing in a day. Note that early in training the first-stage detection output is too noisy to provide stable inputs for the second-stage refinement. To mitigate this issue we used the ground-truth detections for the first 2.5k iterations when training the second-stage network.

We compared the discussed approach to our implementation of IntentNet [5] which we extended to support multiple classes and tuned to obtain better results than reported in the original paper. In addition, using the published results we compared to the recently proposed end-to-end SpAGNN method that takes into account interactions between the traffic actors [6]. We evaluated the methods using both detection and prediction metrics. Following earlier literature for detection metrics, we set the IoU detection matching threshold to 0.7, 0.1, 0.3 for vehicles, pedestrians, and bicyclists, respectively. For prediction metrics we set the probability threshold to obtain a recall of 0.8 as the operational point, as proposed in [6]. In particular, we report average precision (AP) detection metric, as well as displacement error (DE) [41] and cross-track (CT) prediction error at 3 seconds. For the multimodal approaches we report both the min-over-M metrics [11], [17] taking the minimal error over all modes (measuring recall) and the performance of the highest-probability mode (measuring precision).

B. Results

The evaluation results on nuScenes data for vehicles, pedestrians, and bicyclists are summarized in Table I with best prediction results shown in bold, where we compare the proposed MultiXNet to the state-of-the-art methods SpAGNN [6] and IntentNet [5]. Note that, in addition to the baseline IntentNet that uses displacement error (DE) in its loss, we also included a version with equally-weighted AT and CT losses instead. This is an extension of the baseline that uses the idea presented in Sec. III-B.1, which was shown to perform well in our experiments.

We can see that all methods achieved similar detection performance across the board. Comparing the state-of-the-art methods SpAGNN and IntentNet, the latter obtained better prediction accuracy on vehicle actors. The authors of SpAGNN did not provide results on other traffic actors so these results are not included in the table. Moreover, we see that IntentNet with AT/CT losses, corresponding to the model described in Sec. III-A that does not model the uncertainty, achieved comparable DE and CT errors as the original IntentNet with DE loss, with slightly improved results for vehicles and bicyclists. While the improvements are not large, this model allows for different weighting of AT and CT error components. This trade-off is an important feature for deployed models in autonomous driving, where prediction accuracies along these two directions may have different importance (e.g., in merging scenarios AT may be more important, while we may care more about CT in passing

TABLE I: Comparison on nuScenes data using the highest-probability mode, with detection performance evaluated using average precision in % (AP) and prediction using displacement error (DE) and cross-track error (CT) at $3s$ in centimeters; results computed on the best-matching mode (i.e., min-over-M) for multimodal methods shown in parentheses where available

	Vehicles			Pedestrians			Bicyclists		
Method	**AP**	**DE**	**CT**	**AP**	**DE**	**CT**	**AP**	**DE**	**CT**
SpAGNN	-	145.0	-	-	-	-	-	-	-
IntentNet (DE)	61.0	117.4	37.7	64.4	83.5	47.3	30.9	184.6	73.7
IntentNet (AT/CT)	60.3	118.3	37.8	63.4	83.6	46.8	31.8	**173.0**	70.2
MultiXNet	60.6	**105.0** (**104.2**)	**29.7** (**29.1**)	66.1	**80.1**	**43.8**	32.6	203.1	**54.8**

TABLE II: Ablation study of the proposed MultiXNet on ATG4D data; "Unc." denotes uncertainty loss from Sec. III-B.1, "2nd" denotes the refinement stage from Sec. III-B.2, and "Mm." denotes the multimodal loss from Sec. III-B.3

			Vehicles			Pedestrians			Bicyclists		
Unc.	2nd	Mm.	**AP**	**DE**	**CT**	**AP**	**DE**	**CT**	**AP**	**DE**	**CT**
			83.9	90.4	26.0	88.4	61.8	32.9	83.2	51.7	23.5
KL-L			84.1	91.9	22.8	88.2	**57.1**	**30.4**	84.6	49.9	21.1
	✓		84.6	**82.2**	22.2	88.7	63.2	33.2	84.3	51.6	23.8
KL-L	✓		84.4	83.3	20.4	88.4	57.6	30.6	83.9	52.0	21.7
	✓	✓	84.0	82.4 (**81.4**)	22.4 (21.8)	88.5	62.6	33.0	84.2	51.2	23.7
KL-L	✓	✓	84.2	83.1 (82.1)	**20.2** (19.8)	88.4	57.2	30.5	84.6	**48.5**	20.7
KL-G	✓	✓	85.0	84.5 (83.4)	20.6 (**19.7**)	88.7	58.1	31.3	84.7	50.4	**20.6**
NL-L	✓	✓	84.6	85.3 (84.0)	**20.2** (19.8)	88.4	58.4	31.0	83.9	50.4	20.9
NL-G	✓	✓	84.5	88.5 (87.7)	21.1 (20.7)	88.6	59.3	32.0	83.5	51.8	21.0
KL-L	no rot.	✓	84.4	86.9 (86.0)	21.9 (21.2)	88.6	57.3	30.7	84.0	50.4	21.5

scenarios). Lastly, the proposed MultiXNet outperformed the competing methods by a significant margin on all three actor types. Taking only vehicles into account, we see that modeling multimodal trajectories led to improvements when considering the min-over-M mode (result given in parentheses), as well as the highest-probability mode, indicating both better recall and better precision of MultiXNet, respectively.

In Table II we present results of an ablation study of the MultiXNet improvements performed on the larger ATG4D data set, involving the components discussed in Sec. III-B. Note that the first row corresponds to the *IntentNet (AT/CT)* method, while the last row of the first part of the table corresponds to the proposed MultiXNet (KL-L denotes the KL divergence loss using Laplace distribution). We can see that all methods had nearly the same AP, which is not a surprising result since all approaches have identical detection architectures. Focusing on the vehicle actors for a moment, modeling uncertainty led to improvements in the CT error, which decreased by 13%. Introducing the actor refinement using the second-stage network resulted in the largest improvement in the DE, leading to a drop of 11%. Note that such large improvements in DE and CT may translate to significant improvements in the SDV performance.

The last three rows of the upper part of Table II give the performance of different variants of the second-stage model. Similarly to the result given previously, modeling for uncertainty led to a substantial improvement of nearly 10% when it comes to the CT error. This can be explained by the fact that outliers are down-weighted due to their larger variance as shown in equation (3), and thus have less impact during training as compared to the case where the variance is not taken into account. Furthermore, in the next two rows we evaluated the models that output multimodal

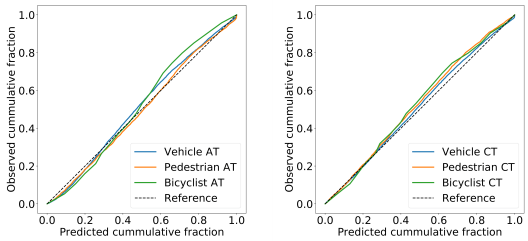

Fig. 3: Reliability diagrams for along-track (AT) (left) and cross-track (CT) (right) dimensions at $3s$ prediction horizon

trajectories. We can see that using the highest-probability mode to measure performance gave comparable results to a unimodal alternative. This is due to a known limitation of such an evaluation scheme, which can not adequately capture the performance of multimodal approaches [11], [17]. For this reason in the parentheses we also report min-over-M, a commonly used multimodal evaluation technique in the literature, which indicated improvements in both DE and CT compared to the other baselines.

Lastly, in the bottom four rows we perform further ablation studies by modifying the uncertainty-aware loss and the second-stage processing. In particular, we analyze a model using the proposed KL divergence that assumes a Gaussian noise model (KL-G) instead of Laplace (KL-L), as well as models that directly optimize negative log-likelihood with Gaussian (NL-G) and Laplace (NL-L) distributions to model the prediction errors. We can see that using KL loss with Laplace distribution as introduced in Sec. III-B.1 resulted in the lowest prediction errors across the three actor types. Moreover, in the last row we show results of MultiXNet where we do not perform rotation of the cropped features as

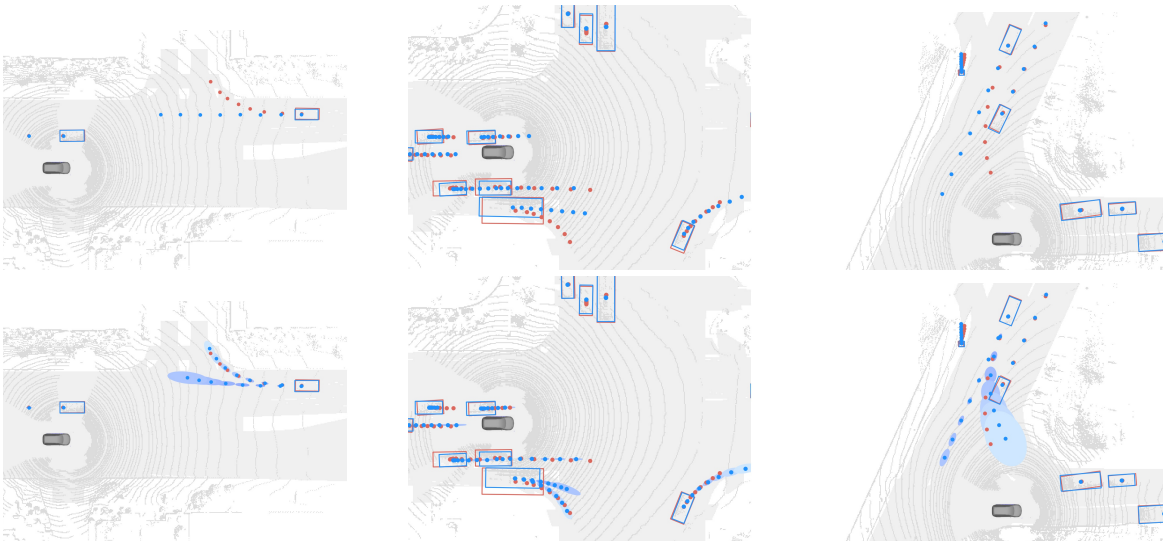

Fig. 4: Qualitative results of the competing models, top row: IntentNet, bottom row: MultiXNet; ground truth shown in red, predictions shown in blue, while colored ellipses indicate one standard deviation of inferred uncertainty for future predictions

described in Sec. III-B.2. We can see that the omission of rotation resulted in a significant drop in performance, further motivating the use of RROI in the second-stage processing.

Let us discuss the results on pedestrians and bicyclists shown in the remainder of Table II. As explained in Sec. III-B.4, we did not use the second-stage refinement nor the multimodal loss for these actors, and the changes indicated in the *2nd* and *Mm.* columns only affected the vehicle branch of the network (results for the same setup changed slightly due to random weight initialization). Similarly to the experiments with vehicles we see that modeling uncertainty led to improved results, with CT improvements between 9% and 13%, as seen in the second, fourth, and sixth rows.

In addition to improved performance, modeling uncertainty also allows reasoning about the inherent noise of future traffic movement. As mentioned in Sec. I, this is an important feature that helps the motion planning generate safe and efficient SDV motions. In Fig. 3 we provide reliability diagrams [10] along the AT and CT dimensions for all three actor types, measured at the prediction horizon of 3 seconds. We can see that the learned uncertainties were well-calibrated, with slight under-confidence for all traffic actors. Bicyclist uncertainties were the least calibrated, followed by pedestrians. As expected, the actor types with the most training data showed the most calibrated uncertainties.

C. Qualitative results

In this section we present several representative case studies, exemplifying the benefits of the proposed MultiXNet over the state-of-the-art IntentNet. Three comparisons of the two methods are shown in Fig. 4, where we do not visualize low-probability MultiXNet trajectories below 0.3 threshold.

In the first case, we see an actor approaching an intersection and making a right-hand turn, where unimodal IntentNet incorrectly predicted that they will continue moving straight through the intersection. On the other hand, MultiXNet predicted a very accurate turning trajectory with high certainty, while also allowing for the possibility of going-straight behavior. Apart from the predictions, we can see that both models detected the two actors in the SDV's surroundings with high accuracy. In the second case, the SDV is moving through an intersection with a green traffic light, surrounded by vehicles. We can see that both models correctly detected and predicted the movements of the majority of the traffic actors. Let us consider motion prediction for a large truck in a right-turn lane on the SDV's right-hand side. Again, IntentNet predicted a straight trajectory while in actuality the actor made a turn. As before, MultiXNet generated multiple modes and provided reasonable uncertainty estimates for both the turning and the going-straight trajectories.

Lastly, the third case shows the SDV in an uncommon three-way intersection. As previously, both models provided accurate detections of the surrounding actors, including one pedestrian at the top of the scene. Let us direct our attention to the vehicle actor approaching the intersection from the upper part of the figure. This actor made an unprotected left turn towards the SDV, which IntentNet mispredicted as going straight. Conversely, we see that MultiXNet produced both possible modes, including a turning trajectory with large uncertainty due to the unusual shape of the intersection.

V. CONCLUSION

We introduced MultiXNet, a multistage model that first infers object detections and predictions, and then refines these predictions using a second stage to output multiple potential future trajectories. The proposed method was evaluated on two large-scale data sets collected on the streets of several cities, where it outperformed the existing state-of-the-art. While obtaining good performance, there are several directions along which the method could be improved. For example, the network does not explicitly model actor interactions, and is lidar- and map-focused while introduction of sensors such as radars and cameras could be very beneficial. These avenues represent areas of our future work.

REFERENCES

[1] P. Stahl, B. Donmez, and G. A. Jamieson, "Anticipation in driving: The role of experience in the efficacy of pre-event conflict cues," *IEEE Transactions on Human-Machine Systems*, vol. 44, no. 5, pp. 603–613, 2014.

[2] ——, "Supporting anticipation in driving through attentional and interpretational in-vehicle displays," *Accident Analysis & Prevention*, vol. 91, pp. 103–113, 2016.

[3] A. Cosgun, L. Ma, *et al.*, "Towards full automated drive in urban environments: A demonstration in gomentum station, california," in *IEEE Intelligent Vehicles Symposium*, 2017, pp. 1811–1818. [Online]. Available: https://doi.org/10.1109/IVS.2017.7995969

[4] W. Luo, B. Yang, and R. Urtasun, "Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net," in *Proc. of the IEEE CVPR*, 2018, pp. 3569–3577.

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

[6] S. Casas, C. Gulino, R. Liao, and R. Urtasun, "Spatially-aware graph neural networks for relational behavior forecasting from sensor data," *arXiv preprint arXiv:1910.08233*, 2019.

[7] G. P. Meyer, J. Charland, S. Pandey, A. Laddha, C. Vallespi-Gonzalez, and C. K. Wellington, "Laserflow: Efficient and probabilistic object detection and motion forecasting," *arXiv preprint arXiv:2003.05982*, 2020.

[8] F.-C. Chou, T.-H. Lin, H. Cui, V. Radosavljevic, T. Nguyen, T.-K. Huang, M. Niedoba, J. Schneider, and N. Djuric, "Predicting motion of vulnerable road users using high-definition maps and efficient convnets," in *IEEE Intelligent Vehicles Symposium (IV)*, 2020.

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

[10] N. Djuric, V. Radosavljevic, H. Cui, T. Nguyen, F.-C. Chou, T.-H. Lin, and J. Schneider, "Uncertainty-aware short-term motion prediction of traffic actors for autonomous driving," in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020.

[11] H. Cui, V. Radosavljevic, F.-C. Chou, T.-H. Lin, T. Nguyen, T.-K. Huang, J. Schneider, and N. Djuric, "Multimodal trajectory predictions for autonomous driving using deep convolutional networks," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 2090–2096.

[12] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 697–12 705.

[13] G. P. Meyer, A. Laddha, E. Kee, C. Vallespi-Gonzalez, and C. K. Wellington, "Lasernet: An efficient probabilistic 3d object detector for autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 677–12 686.

[14] G. P. Meyer, J. Charland, D. Hegde, A. Laddha, and C. Vallespi-Gonzalez, "Sensor fusion for joint 3d object detection and semantic segmentation," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

[15] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1907–1915.

[16] Y. Zhou, P. Sun, Y. Zhang, D. Anguelov, J. Gao, T. Ouyang, J. Guo, *et al.*, "End-to-end multi-view fusion for 3d object detection in lidar point clouds," *arXiv preprint arXiv:1910.06528*, 2019.

[17] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. Torr, and M. Chandraker, "Desire: Distant future prediction in dynamic scenes with interacting agents," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 336–345.

[18] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Multi-agent generative trajectory forecasting with heterogeneous data for control," *arXiv preprint arXiv:2001.03093*, 2020.

[19] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social gan: Socially acceptable trajectories with generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2255–2264.

[20] T. Zhao, Y. Xu, M. Monfort, W. Choi, C. Baker, Y. Zhao, Y. Wang, and Y. N. Wu, "Multi-agent tensor fusion for contextual trajectory prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 126–12 134.

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

[22] V. Kosaraju, A. Sadeghian, R. Martín-Martín, I. Reid, H. Rezatofighi, and S. Savarese, "Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks," in *Advances in Neural Information Processing Systems*, 2019, pp. 137–146.

[23] Y. Ma, X. Zhu, S. Zhang, R. Yang, W. Wang, and D. Manocha, "Trafficpredict: Trajectory prediction for heterogeneous traffic-agents," in *AAAI Conference on Artificial Intelligence*, 2019.

[24] R. Chandra, U. Bhattacharya, A. Bera, and D. Manocha, "Traphic: Trajectory prediction in dense and heterogeneous traffic using weighted interactions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8483–8492.

[25] Y. Yuan and K. Kitani, "Diverse trajectory forecasting with determinantal point processes," *arXiv preprint arXiv:1907.04967*, 2019.

[26] E. Wang, H. Cui, S. Yalamanchi, M. Moorthy, F.-C. Chou, and N. Djuric, "Improving movement predictions of traffic actors in bird's-eye view models using GANs and differentiable trajectory rasterization," in *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2020.

[27] T. Phan-Minh, E. C. Grigore, F. A. Boulton, O. Beijbom, and E. M. Wolff, "Covernet: Multimodal behavior prediction using trajectory sets," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 14 074–14 083.

[28] H. Cui, T. Nguyen, F.-C. Chou, T.-H. Lin, J. Schneider, D. Bradley, and N. Djuric, "Deep kinematic models for physically realistic prediction of vehicle trajectories," *2020 International Conference on Robotics and Automation (ICRA)*, 2020.

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

[30] A. Jain, S. Casas, R. Liao, Y. Xiong, S. Feng, S. Segal, and R. Urtasun, "Discrete residual flow for probabilistic pedestrian behavior prediction," *arXiv preprint arXiv:1910.08041*, 2019.

[31] W. Zeng, W. Luo, S. Suo, A. Sadat, B. Yang, S. Casas, and R. Urtasun, "End-to-end interpretable neural motion planner," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8660–8669.

[32] P. Wu, S. Chen, and D. Metaxas, "Motionnet: Joint perception and motion prediction for autonomous driving based on bird's eye view maps," *arXiv preprint arXiv:2003.06754*, 2020.

[33] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in *ICCV*, 2017.

[34] C. Gong and D. McNally, "A methodology for automated trajectory prediction analysis," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2004, p. 4788.

[35] G. P. Meyer, "An alternative probabilistic interpretation of the huber loss," *arXiv preprint arXiv:1911.02088*, 2019.

[36] G. P. Meyer and N. Thakurdesai, "Learning an uncertainty-aware object detector for autonomous driving," *arXiv preprint arXiv:1910.11375*, 2019.

[37] M. Liang, B. Yang, Y. Chen, R. Hu, and R. Urtasun, "Multi-task multi-sensor fusion for 3d object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7345–7353.

[38] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 621–11 631.

[39] A. Paszke, S. Gross, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035.

[40] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[41] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 961–971.