IntentNet: Learning to Predict Intention from Raw Sensor Data

Sergio Casas, Wenjie Luo, Raquel Urtasun

Uber Advanced Technologies Group, University of Toronto
{sergio.casas,wenjie,urtasun}@uber.com

Abstract: In order to plan a safe maneuver, self-driving vehicles need to understand the intent of other traffic participants. We define intent as a combination of discrete high level behaviors as well as continuous trajectories describing future motion. In this paper we develop a one-stage detector and forecaster that exploits both 3D point clouds produced by a LiDAR sensor as well as dynamic maps of the environment. Our multi-task model achieves better accuracy than the respective separate modules while saving computation, which is critical to reduce reaction time in self-driving applications.

Keywords: Deep Learning, Self-Driving Cars, Perception, Prediction

1 Introduction

Autonomous driving is one of the most exciting problems of modern artificial intelligence. Self-driving vehicles have the potential to revolutionize the way people and freight move. While a plethora of systems have been built in the past few decades, many challenges still remain. One of the fundamental difficulties is that self driving vehicles have to share the roads with human drivers, which can perform maneuvers that are difficult to predict.

Human drivers understand intention by exploiting the actors' past motion as well as prior knowledge about the scene (e.g. the location of lanes, direction of driving). Previous work [1, 2, 3] attempted to solve this problem by first performing vehicle detection and then extracting intent from the position and motion of detected bounding boxes. This, however, restricts the information that the intent estimation module has access to, resulting in suboptimal estimates. Very recently, FaF [4] directly exploited LiDAR sensor data to predict the future trajectories of vehicles. However, the trajectories were only predicted for 1 second into the future and no intent prediction was done beyond motion estimation.

In this paper, we take this approach one step further and propose a novel deep neural network that reasons about both high level behavior and long term trajectories. Inspired by how humans perform this task, we design a network that exploits motion and prior knowledge about the road topology in the form of maps containing semantic elements such as lanes, intersections and traffic lights. In particular, our *IntentNet* is a fully-convolutional neural network that outputs three types of variables in a single forward pass corresponding to: detection scores for vehicle and background classes, high level action probabilities corresponding to discrete intention, and bounding box regressions in the current and future time steps to represent the intended trajectory. Our architecture allows us to jointly optimize all tasks, fixing the distribution mismatch problem between tasks when solving them separately. Importantly, our design also enables the system to propagate uncertainty through the different components. In addition, our approach is computationally efficient by design as all tasks share the heavy neural network feature computation.

We demonstrate the effectiveness of our approach in the tasks of detection and intent prediction by showing that our system surpasses other real-time, state-of-the art detectors while outperforming previous intent prediction approaches, both in its continuous and discrete counterparts. In the remainder of the paper, we first discuss related work and then present our model followed by experimental evaluation and conclusion.

2nd Conference on Robot Learning (CoRL 2018), Zürich, Switzerland.

2 Related Work

In this section we first review recent advances in object detection, focusing on single-stage and 3D object detection. We then discuss motion and intent prediction approaches.

Object detection: Many proposal based approaches [5, 6] have been developed after the seminal work of RCNN [7]. While these methods perform really well, they are typically not suitable for real-time applications as they are computationally demanding. In contrast, single stage detectors [8, 9] provide a more efficient solution. YOLO [8] breaks down the image into grids and makes multi-class and multi-scale predictions at each cell. SSD [9] added the notion of anchor boxes, which reduces object variance in size and pose. RetinaNet [10] showed that single-stage detectors can outperform two-stage detectors in both speed and accuracy. Geiger *et al.* [11] improved the ability to estimate object orientation by jointly reasoning about the scene layout. More recently, Vote3Deep [12] proposed to voxelize point clouds and exploit 3D CNNs [13]. Subsequently, FaF [4] and PIXOR [14] exhibited superior performance in terms of speed and accuracy by exploiting a bird's eye view representation. Additionally, [4] aggregated several point clouds from the past. Approaches that use the projection representation (VeloFCN [15], MV3D [16]) or handle point clouds directly (PointNet [17]) have also been proposed. However, these methods suffer from either limited performance or heavy computation and thus are not suitable for self driving [18]. Liang *et al.* [19] proposed a real time 3D detector that exploits multiple sensors (i.e., camera and LiDAR).

Motion Forecasting: This refers to the task of predicting future locations of an actor given current and past information. DESIRE [20] introduced an RNN encoder-decoder framework in the setting of multiple interacting agents in dynamic scenes. Ma *et al.* [21] proposed to couple game theory and deep learning to model pedestrians interactions and estimate person-specific behavior parameters. Ballan *et al.* [22] exploited the interplay between the dynamics of moving agents and the semantics of the scene for scene-specific motion prediction. Soo *et al.* [23] created an EgoRetinal map to predict plausible future ego-motion trajectories in egocentric stereo images. Hoermann et al. [24] utilized a dynamic occupancy grid map as input to a deep convolutional neural network to perform long-term situation prediction in autonomous driving. SIMP [3] parametrized the output space as insertion areas where the vehicle of interest could go, predicting an estimated time of arrival and a spatial offset. Djuric *et al.* [25] rasterized representations of each actor's vicinity in order to predict their future motion. FaF [4] pioneered the unification of detection and short term motion forecasting from LiDAR point clouds in driving scenarios.

Intention Prediction: The intention of an actor can be seen as the sequence of actions it will take in order to achieve an objective. Fathi *et al.* [26] developed a probabilistic generative model to predict daily actions using gaze. In the context of intelligent vehicles, Zhang *et al.* [27] proposed to model high level semantics in the form of traffic patterns through a generative model that also reasons about the geometry and objects present in the scene. Jain *et al.* [28] proposed an autoregressive HMM to anticipate driving maneuvers a few seconds before they occur by exploiting video from a face and a rear camera together with features from the map. Streubel *et al.* [1] utilized HMMs to estimate the direction of travel while approaching a 4-way intersection. Kim *et al.* [29] predicted egocentric intention of lane changes. Recently, Phillips *et al.* [2] utilized LSTMs to predict cars' intention at generalizable intersections. SIMP [3] suggested to model intention implicitly by defining a discrete set of insertion areas belonging to particular lanes. Unfortunately, most of the work in this area lacks solid evaluation. For instance, [2] used 1 hour of driving data across only 9 intersections for both training and evaluation (cross-validated), while [3] apply the SIMP framework to only 640 meters of highway.

IntentNet is inspired by FaF [4], which performs joint detection and future prediction. IntentNet achieves a more accurate vehicle detection and trajectory forecasting, enlarges the prediction horizon and estimates future high-level driver's behavior. The key contributions to the performance gain are (i) a more suitable architecture based on an early fusion of a larger number of previous LiDAR sweeps, (ii) a parametrization of the map that allows our model to understand traffic constraints for all vehicles at once and (iii) an improved loss function that includes a temporal discount factor to account for the inherent ambiguity of the future.

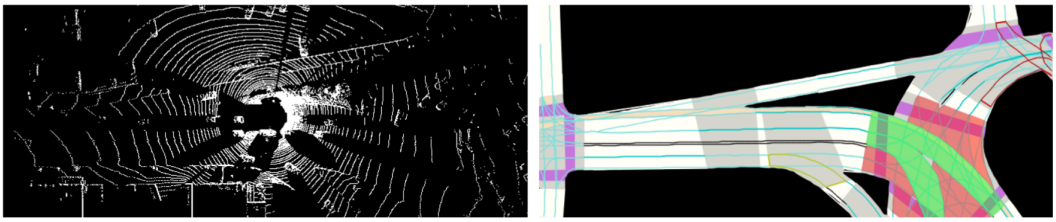

Figure 1: Input parametrization. Left: Voxelized LiDAR in BEV (height aggregated in a single channel for visualization purposes). Right: Rasterized map (in RGB for visualization purposes)

3 Learning to Predict Intention

In this section, we present our approach to jointly detect vehicles and predict their intention directly from raw sensor data. Towards this goal, we exploit 3D LiDAR point clouds and dynamic HD maps containing semantic elements such as lanes, intersections and traffic lights. In the following, we describe our parametrization, network architecture, learning objective and inference procedure.

3.1 Input parametrization

3D point cloud: Standard convolutional neural networks (CNNs) perform discrete convolutions, assuming a grid structured input. Following [16, 4, 14], we represent point clouds in bird's eye view (BEV) as a 3D tensor, treating height as our channel dimension. This input parametrization has several key advantages: (i) computation efficiency due to dimensionality reduction (made possible as vehicles drive on the ground), (ii) non-overlapping targets (contrary to camera-view representations, where objects can overlap), (iii) preservation of the metric space (undistorted view) that eases the creation of priors regarding vehicle sizes, and (iv) this representation also makes the fusion of LiDAR and map features trivial as both are defined in bird's eye view. We utilize multiple consecutive LiDAR sweeps (corrected by ego-motion) as the past is fundamental to accurately estimate both intention and motion forecasting. We diverge from previous work and stack together height and time dimensions into the channel dimension as this allows us to use 2D convolutions to fuse time information. As shown in our experiments, this is more effective than the non-padded 3D convolutions proposed in [4]. This gives us a tensor of size: $(\frac{L}{\Delta L}, \frac{W}{\Delta W}, \frac{H}{\Delta H} \cdot T)$, where L, W and H are the longitudinal, transversal and normal physical dimensions of the scene; ΔL, ΔW and ΔH are the voxel sizes in the corresponding directions and T is the number of LiDAR sweeps.

Dynamic maps: We form a BEV representation of our maps by rasterization. We exploit static information including roads, lanes, intersections, crossings and traffic signs, in conjunction with traffic lights, which contain dynamic information that changes over time (i.e., traffic light state changes between green, yellow and red). We represent each semantic component in the map with a binary map (i.e., 1 or -1). Roads and intersections are represented as filled polygons covering the whole drivable surface. Lane boundaries are parametrized as poly-lines representing the left and right boundaries of lane segments. Note that we use three binary masks to distinguish lane types, as lane boundaries can be crossed or not, or only in certain situations. Lane surfaces are rasterized to distinguish between straight, left and right turns, as this information is helpful for intention prediction. We also use two extra binary masks for bike and bus lanes as a way to input a prior of non-drivable road areas. Furthermore, traffic lights can change the drivable region dynamically. We encode the state of the traffic light into the lanes they govern. We rasterize the surface of the lane succeeding the traffic light in one out of three binary masks depending on its state: green, yellow or red. One extra layer is used to indicate whether those lanes are protected by its governing traffic light, i.e. cars in other lanes must yield. This situation happens in turns when the arrow section of the traffic light is illuminated. We estimate the traffic light states using cameras in our self-driving vehicle. We also infer the state of some unobserved traffic lights that directly interact with known traffic light states. For example, a straight lane with unknown traffic light state that collides with a protected turn with green traffic light state can be safely classified as being red. Lastly, traffic signs are also encoded into their governed lane segments, using two binary masks to distinguish between yields and stops. In total, there are 17 binary masks used as map features, resulting in a 3D tensor that represents the

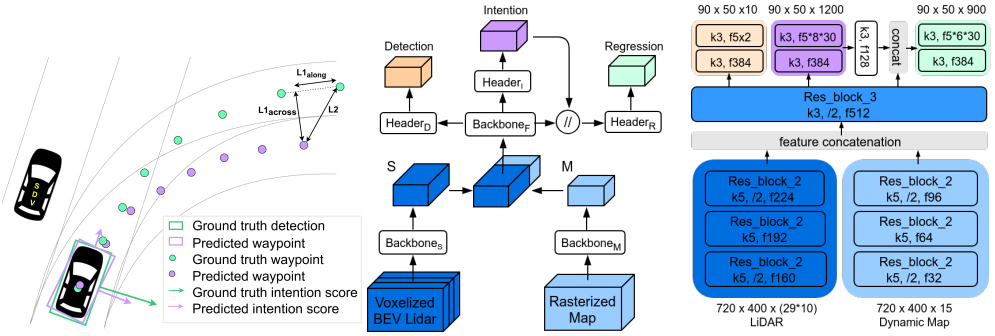

Figure 2a: Output parametrization Figure 2b: Architecture overview Figure 2c: Layer details

map. Fig. 1 shows an example, where different elements (e.g., lane markers in cyan, crossings in magenta, alpha blended traffic lights with their state colored) are depicted.

3.2 Output parametrization

Our model predicts drivers' intentions in both discrete and continuous form.

Trajectory regression: For each detected vehicle, we parametrize its trajectory as a sequence of bounding boxes, including current and future locations. Assuming cars are non-deformable objects, we treat their size (w, h) as a constant estimated by the detector. The pose in each time stamp is 3D and contains the bounding box center (c_x^t, c_y^t) and heading ϕ^t of the vehicle in BEV coordinates (see Fig. 2a).

High level actions: We frame the discrete intention prediction problem as a multi-class classification with 8 classes: *keep lane, turn left, turn right, left change lane, right change lane, stopping/stopped, parked and other*, where *other* can be any other action such as reversed driving.

3.3 Network architecture

Work across different domains [30, 31, 32] has shown that late fusion delivers stronger performance than early fusion. IntentNet exploits a late fusion of LiDAR and map information through an architecture consisting of a two-stream backbone network and three task-specific branches on top (see Figs. 2b and 2c).

Backbone network: Our single-stage model takes two 3D tensors as input: the voxelized BEV LiDAR and the rasterization of our dynamic maps. We utilize a two-stream backbone network, where two different 2D CNNs process each data stream separately. The feature maps obtained from those subcomponents are then concatenated along the depth dimension and fed to the fusion subnetwork. We use a small downsampling coefficient in our network of 8x since each vehicle represents a small set of pixels in BEV, e.g., when using a resolution of 0.2 m/pixel, a car on average occupies 18 x 8 pixels. To provide accurate long term intention prediction and motion forecasting, the network needs to extract rich motion information from the past and geometric details of the scene together with traffic rule information. Note that vehicles typically drive at 50 km/h in urban scenes, traversing 42 meters in only 3 seconds. Thus we need our network to have a sufficiently large effective receptive field [33] to extract the desired information. To keep both coarse and fine grained features, we exploit residual connections [34]. We refer the reader to Fig. 2b for more details of our network architecture.

Header network: The header network is composed of three task specific branches that take as input the shared features from the backbone network. The detection branch outputs two scores for each anchor box at each feature map location, one for vehicle and one for background. An anchor is a predefined bounding box with orientation that serves as a prior for detection. Similar to [4, 5, 9], we use multiple anchors for each feature map location. The intention network performs

a multi-class classification over the set of high level actions, assigning a calibrated probability to the 8 possible behaviors at each feature map location. The discrete intention scores are in turn fed into an embedding convolutional layer to provide extra features to condition the motion estimation. The motion estimation branch receives the concatenation of the shared features and the embedding from the high level action scores, and outputs the predicted trajectories for each anchor box at each feature map location.

3.4 Learning

Our model is fully differentiable and thus can be trained end-to-end through back-propagation. In particular, we minimize a multi-task loss containing a regression term for the trajectory prediction over T time steps, a binary classification term for the detection (background vs vehicle) and a multi-class classification for discrete intention. Thus

$$\mathcal{L}(\theta) = \mathcal{L}_{cla}(\theta) + \alpha \cdot \sum_{t=0}^{T} \lambda^t \cdot \mathcal{L}^t_{int}(\theta) + \beta \cdot \sum_{t=0}^{T} \lambda^t \cdot \mathcal{L}^t_{reg}(\theta)$$

where $t = 0$ is the current frame and $t > 0$ the future, θ the model parameters and λ a temporal discount factor to ensure distance times into the future do not dominate the loss as they are more difficult to predict. We mow define the loss functions we employ in more details.

Detection: We define a binary focal loss [10], computed over all feature map locations and predefined anchor boxes, assigned using the matching strategy proposed in [9]:

$$\mathcal{L}_{cla}(\theta) = \textstyle\sum_{i,j,k} -(1-\overline{p}_{i,j,k;\theta}) \cdot \log \overline{p}_{i,j,k;\theta}, \qquad \overline{p}_{i,j,k;\theta} = \begin{cases} p_{i,j,k;\theta} & \text{if} \quad q_{i,j,k} = 1, \\ 1 - p_{i,j,k;\theta} & \text{otherwise} \end{cases}$$

where i, j are the location indices on the feature map and k is the index over the predefined set of anchor boxes; $q_{i,j,k}$ is the class true label and $p_{i,j,k;\theta}$ the predicted probability. We define as positive samples, i.e. $q_{i,j,k} = 1$, those predefined anchor boxes having an associated ground truth box. In particular, for each anchor box, we find the ground truth box with the biggest intersection over union (IoU). If the IoU is bigger than a threshold of 0.5, we assign 1 to its corresponding label $q_{i,j,k}$. In case there is a ground truth box that has not been assigned to any anchor box, we assign it to the highest overlapping anchor box ignoring the threshold. Due to the imbalance of positive and negative samples we have found it helpful to not only use focal loss but also to apply hard negative mining during training. Thus, we rank all negative samples by their predicted score $p_{i,j,k}$ and take the top negative samples with a ratio of 3:1 with respect to the number of positive samples.

Trajectory regression: We frame the detector regression targets as a function of the dimensions of their associated anchor boxes. As we reason in BEV, all objects have similar size as no perspective effect is involved. Thus, we can exploit object shape priors and make anchor boxes similar to real object sizes. This helps reduce the variance of the regression targets, leading to better training. In particular, we define

$$\overline{c^t_x} = \tfrac{c^t_x - c^{anchor}_x}{w^{anchor}} \qquad \overline{\phi^t_{sin}} = \sin \phi^t \qquad \overline{w} = \log \tfrac{w}{w^{anchor}}$$

$$\overline{c^t_y} = \tfrac{c^t_y - c^{anchor}_y}{h^{anchor}} \qquad \overline{\phi^t_{cos}} = \cos \phi^t \qquad \overline{h} = \log \tfrac{h}{h^{anchor}}$$

We apply a weighted smooth L1 loss to the regression targets associated to the positive samples only. Note that for the future time steps ($t \,\epsilon\, [1, T-1]$), the target set $\mathcal{R}_t$ does not include the bounding box size ($\overline{w}$, $\overline{h}$), which are only predicted at the current time step ($t = 0$).

$$\mathcal{L}^t_{reg}(\theta) = \sum_{r \epsilon \mathcal{R}_t} \chi_r \cdot l^t_{r;\theta}, \qquad l^t_{r;\theta} = \begin{cases} 0.5 \cdot (x^t_{r;\theta} - y^t_r)^2, & \text{if } |x^t_{r;\theta} - y^t_r| < 1 \\ |x^t_{r;\theta} - y^t_r| - 0.5, & \text{otherwise} \end{cases}$$

where t is the prediction time step, $x^t_{r;\theta}$ refers to the predicted value of the r-th regression target, y^t_r is the ground truth value of such regression target, and χ_r is the weight assigned to the r-th regression target.

Model	Detection mAP @ IOU				
	0.5	0.6	0.7	0.8	0.9
SqueezeNet	74.0	62.3	41.9	13.8	0.2
SSD	84.0	75.1	58.2	26.0	1.0
MobileNet	86.1	78.3	60.4	27.5	1.1
FaF	89.8	82.5	68.1	35.8	2.5
FaF'	88.4	80.1	64.1	30.9	1.6
IntentNet	**94.4**	**89.4**	**75.4**	**43.5**	**3.9**

Table 1: Detection performance with objects containing $p \geq 1$ LiDAR points

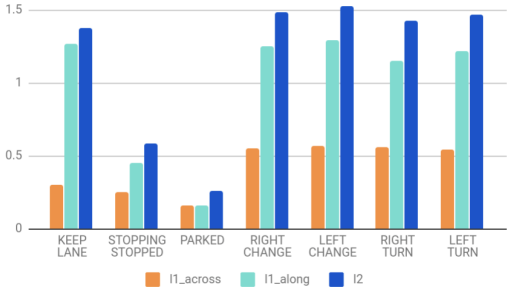

Figure 3: L1 error along and across track and L2 error split by ground truth discrete intention

Intention prediction: We employ a cross entropy loss over the set of high level actions. To address the high imbalance in the intention distribution, we downsample the dominant classes $\{$*keep lane, stopping/stopped and parked*$\}$ by 95%. We found this strategy to work better than re-weighting the loss by the inverse frequency in the training set. Note that we do not discard those examples for detection and trajectory regression.

3.5 Inference

During inference, IntentNet produces 3D detections with their respective continuous and discrete intentions for all vehicles in the scene in a single forward pass. Preliminary detections are extracted by applying a threshold of 0.1 to the classification probabilities, with the intention of achieving high recall. From these feature map locations, we examine the regression targets and anchor boxes, and use NMS to de-duplicate detections. Since our model can predict future waypoints for each vehicle, it provides a strong prior for associating detections among different time steps. At any time step, we have a detection from the current forward pass and predictions produced at previous time steps. Therefore, by comparing the current location against past predictions of the future, we decode tracklets for each vehicle in the scene. This simple tracking system proposed in FaF [4] also allows us to recover missing false negatives and discard false positives by updating the classification scores based on previous predictions.

4 Experimental Evaluation

We evaluate our approach in the tasks of detection, intention prediction and motion forecasting. To enable this, we collected a large scale dataset from a roof-mounted LiDAR on top of a self-driving vehicle driving around several cities in North America. It contains over 1 million frames collected from over 5,000 different scenarios, which are sequences of 250 frames captured sequentially at 10 Hz. Our labels are tracklets of 3D non-axis align bounding boxes. The dataset is highly imbalanced and thus challenging, containing the following number of examples of each behavior; *keep lane*: 10805k, *turn left*: 546k, *turn right*: 483k, *left change lane*: 290k, *right change lane*: 224k, *stopping/stopped*: 12766k, *parked*: 29110k and *others*: 100k.

Implementation details: We use a birds eye view (BEV) region with $L = 144$, $W = 80$ meters from the center of the autonomous vehicle and $H = 5.8$ meters from the ground, for both training and evaluation. We set the resolution to be $\Delta L = \Delta W = \Delta H = 0.2$ meters. We use $T = 10$ past LiDAR sweeps to provide enough context in order to perform accurate long term prediction of 3 seconds. Thus, our input is a 3D tensor of shape $(29 \cdot 10, 720, 400)$. We use 5 predefined anchor boxes of size 3.2 meters and $1:1, 1:2, 2:1, 1:6, 6:1$, and a threshold of 0.1 for the classification score to consider a detection positive during inference. We train our model from scratch using Adam optimizer [35], a learning rate of 1e-4 and a weight decay of 1e-4. We employ a temporal discount factor $\lambda = 0.97$ for our future predictions. We used a batch of size 6 for each GPU and perform distributed training on 8 Nvidia 1080 GPUs for around 24h.

Model	L1 error along track (m)				L1 error across track (m)				L1 error heading (deg)			
	0s	1s	2s	3s	0s	1s	2s	3s	0s	1s	2s	3s
FaF	0.28	0.53	-	-	0.17	0.26	-	-	6.06	6.47	-	-
FaF'	0.29	0.49	0.95	1.67	0.21	0.29	0.45	0.69	6.35	6.57	7.16	8.02
IntentNet*	0.27	0.47	0.92	1.64	0.18	0.27	0.43	0.65	5.81	6.09	6.69	7.62
IntentNet	**0.26**	**0.46**	**0.91**	**1.61**	**0.15**	**0.21**	**0.34**	**0.53**	**5.14**	**5.35**	**5.83**	**6.54**

Table 2: Regression error computed over the intersection of true positive detections from the four models. IntentNet* is our model without map and without high level actions

Model	Metric	P	S	LK	TR	TL	LCR	LCL	Others	Mean
MLP	Acc	63.4	68.5	79.4	98.8	98.6	100	100	100	88.6
	F1	67.2	39.0	32.5	00.0	00.0	00.0	00.0	00.0	17.3
LSTM	Acc	66.9	69.2	79.9	99.2	98.7	100	100	100	89.3
	F1	69.8	48.1	37.8	00.0	00.0	00.0	00.0	00.0	19.5
IntentNet w/o map	Acc	93.9	89.5	89.5	97.5	96.6	98.3	97.9	97.5	95.1
	F1	98.7	80.4	80.4	51.9	52.1	18.2	24.1	21.6	53.4
IntentNet	Acc	**99.2**	**94.9**	**93.5**	**98.6**	**98.6**	**99.2**	**99.1**	**98.5**	**97.7**
	F1	**98.8**	**91.0**	**88.7**	**66.4**	**73.1**	**50.3**	**55.5**	**45.2**	**71.1**

Table 3: Average intention prediction performance at the current timestep

Detection results: We compare mean average precision (mAP) at different IoU levels with other object detectors that can also run inference in real-time (i.e., less than 100ms) including SqueezeNet [36], SSD [9], MobileNet [37], FaF [4] and FaF' (adaptation of FaF where the motion forecasting header is trivially extended to predict 3 seconds). As shown in Table 1, our model is clearly superior across all IoU levels. Note that all models use the same anchor boxes and downsampling factor for the comparison to be consistent.

Trajectory regression results: To the best of our knowledge, FaF [4] is the only previous work that performs joint detection and motion forecasting from LiDAR. As shown in Table 2 IntentNet outperforms both FaF and FaF' in both along and across track errors as well as heading. The performance leap is a combination of network architecture improvements and the usage of prior knowledge in the form of maps. To be fair, the metrics are reported over the intersection of true positives of all the four models, which represents over 90 % of the validation set. In addition, Fig. 3 depicts regression metrics split by high level action. We highlight that IntentNet is able to learn complex velocity profiles such as the ones in turns and lane changes, keeping the along track error almost as low as in lane keeping.

Intention prediction: We compare our approach with the models proposed in [2], adapting their classifiers to perform our task, which is a generalization of theirs. Phillips *et al.* extract several handcrafted features including *base features* such as velocity, acceleration, number of lanes to the curb and the median and headway distance to preceding vehicle, at the current time step; *history features* containing such information for previous time steps; *traffic features* containing up to six neighbours base features and *rule features* such as whether the car can turn left/right from its lane. We then train their model in our dataset. Table 3 shows the results of their two best performers models: an MLP and an LSTM. Our model clearly outperforms the baselines, especially on the less represented high level actions where the baselines are unable to recognize turns or lane changes. Note that we also tried applying the same downsampling method we use to the baselines but it resulted in worse overall performance and therefore was omitted here. As shown, IntentNet is able to generalize despite the extreme imbalance in the behavior distribution.

Ablation study: We conducted an ablation study in order to evaluate how much each of the contributions proposed in this paper helped towards achieving the final results, which is shown in Table 4. Using a 2D CNN with early fusion of the different LiDAR sweeps delivers a much robust detector,

Conv type 3D	Conv type 2D	Context 0.5s	Context 1.0s	M	I	mAP@0.5 $p \geq 1$	mAP@0.5 $p \geq 5$	mAP@0.7 $p \geq 1$	mAP@0.7 $p \geq 5$	Recall	L2@0s (m)	L2@3s (m)
✓		✓				88.2	92.0	62.9	68.3	91.3	0.37	2.16
	✓	✓				91.9	94.8	71.7	76.8	93.7	0.33	1.90
	✓		✓			92.4	95.0	72.8	77.5	94.1	0.33	1.85
	✓		✓		✓	90.1	92.5	69.8	74.4	93.8	0.34	1.84
	✓		✓	✓		94.3	**96.3**	**75.5**	**79.6**	95.8	**0.33**	1.80
	✓		✓	✓	✓	**94.4**	96.2	75.4	**79.6**	**95.9**	**0.33**	**1.79**

Table 4: Ablation study of IntentNet different contributions. *M* column states whether the model uses the map or not. *I* column states if discrete intention is being predicted.

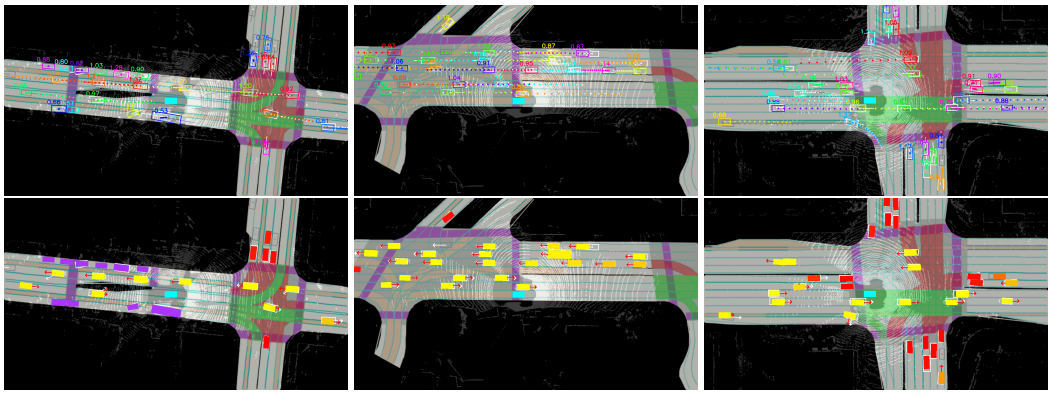

Figure 4: Qualitative results. Ground truth is displayed in white (grey if the ground truth did not have LiDAR points) and predictions in color. Legend for high level actions - *parked*: purple box, *stopping/stopped*: red box, *keep lane*: straight arrow, *turns*: 90 ° arrow, *lane changes*: 30 ° arrow. For the arrows, the longer the higher the probability. *Others* class is not shown for simplicity.

being able to understand the time dimension better than a 3D CNN as proposed in [4]. Increasing the context from 0.5 seconds to 1 second (5 to 10 input LiDAR sweeps) gives us a small gain, decreasing the long term L2 error. Notice that even though the detector is able to have higher recall while using the same confidence threshold (0.1), the regressions become better in average (even when taking into account those harder examples). Adding the loss for the discrete intention estimation degrades the detector/regressor performance due to the fact that it is very hard to predict behavior purely based on motion. However, after adding the map, the system is able to predict the high level behavior of vehicles and thus adding the loss improves general performance. For a direct comparison to FaF [4], its detection results (mAP) are 89.8 (IOU=0.5,p$\geq$1), 93.3 (IOU=0.5,p$\geq$5), 68.1 (IOU=0.7,p$\geq$1) and 73.4 (IOU=0.7,p$\geq$5); its recall is 92.9% and its L2@0s is 0.36 meters.

Qualitative results: Fig. 4 shows our results in the 144 x 80 meters region. We can see 3 pairs of frames belonging to different scenarios. We display the detections and continuous intent prediction in the top row and discrete intent prediction in the bottom row. Our model is able to predict lane changes and detect big size vehicles (left), have a high precision and recall in cluttered scenes (center) and predict turns (right).

5 Conclusion

In this paper we introduce IntentNet, a learnable end-to-end model that is able to tackle the tasks of detection and intent prediction of vehicles in the context of self-driving cars. By exploiting 3D point clouds produced by a LiDAR sensor and prior knowledge of the scene coming from an HD map, we are able to achieve higher performance than previous work across all tasks, with a single neural network. In the future, we plan to investigate how more sophisticated algorithms can model the statistical dependencies between discrete and continuous intention. We also plan to extend our approach to handle pedestrians and bicyclists.

References

[1] T. Streubel and K. H. Hoffmann. Prediction of driver intended path at intersections. In *Intelligent Vehicles Symposium Proceedings, 2014 IEEE*, 2014.

[2] D. J. Phillips, T. A. Wheeler, and M. J. Kochenderfer. Generalizable intention prediction of human drivers at intersections. In *Intelligent Vehicles Symposium (IV), 2017 IEEE*. IEEE, 2017.

[3] Y. Hu, W. Zhan, and M. Tomizuka. Probabilistic prediction of vehicle semantic intention and motion. *arXiv preprint arXiv:1804.03629*, 2018.

[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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[5] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, 2015.

[6] J. Dai, Y. Li, K. He, and J. Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems*, 2016.

[7] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014.

[8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[9] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, 2016.

[10] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. *arXiv preprint arXiv:1708.02002*, 2017.

[11] A. Geiger, C. Wojek, and R. Urtasun. Joint 3d estimation of objects and scene layout. In *Advances in Neural Information Processing Systems*, 2011.

[12] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner. Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017.

[13] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Computer Vision (ICCV), 2015 IEEE International Conference on*. IEEE, 2015.

[14] B. Yang, W. Luo, and R. Urtasun. Pixor: Real-time 3d object detection from point clouds. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[15] B. Li, T. Zhang, and T. Xia. Vehicle detection from 3d lidar using fully convolutional network. *arXiv preprint arXiv:1608.07916*, 2016.

[16] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia. Multi-view 3d object detection network for autonomous driving. In *IEEE CVPR*, 2017.

[17] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proc. Computer Vision and Pattern Recognition (CVPR)*, 2017.

[18] M. Simon, S. Milz, K. Amende, and H.-M. Gross. Complex-yolo: Real-time 3d object detection on point clouds. *arXiv preprint arXiv:1803.06199*, 2018.

[19] M. Liang, B. Yang, S. Wang, and R. Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[20] 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. 2017.

[21] W.-C. Ma, D.-A. Huang, N. Lee, and K. M. Kitani. Forecasting interactive dynamics of pedestrians with fictitious play. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE, 2017.

[22] L. Ballan, F. Castaldo, A. Alahi, F. Palmieri, and S. Savarese. Knowledge transfer for scene-specific motion prediction. In *European Conference on Computer Vision*, 2016.

[23] H. Soo Park, J.-J. Hwang, Y. Niu, and J. Shi. Egocentric future localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[24] S. Hoermann, M. Bach, and K. Dietmayer. Dynamic occupancy grid prediction for urban autonomous driving: A deep learning approach with fully automatic labeling. *arXiv preprint arXiv:1705.08781*, 2017.

[25] N. Djuric, V. Radosavljevic, H. Cui, T. Nguyen, F.-C. Chou, T.-H. Lin, and J. Schneider. Motion prediction of traffic actors for autonomous driving using deep convolutional networks. *arXiv preprint arXiv:1808.05819*, 2018.

[26] A. Fathi, Y. Li, and J. M. Rehg. Learning to recognize daily actions using gaze. In *European Conference on Computer Vision*, 2012.

[27] H. Zhang, A. Geiger, and R. Urtasun. Understanding high-level semantics by modeling traffic patterns. In *Proceedings of the IEEE international conference on computer vision*, 2013.

[28] A. Jain, H. S. Koppula, B. Raghavan, S. Soh, and A. Saxena. Car that knows before you do: Anticipating maneuvers via learning temporal driving models. In *Proceedings of the IEEE International Conference on Computer Vision*, 2015.

[29] I.-H. Kim, J.-H. Bong, J. Park, and S. Park. Prediction of drivers intention of lane change by augmenting sensor information using machine learning techniques. *Sensors*, 2017.

[30] R. Zhang, S. A. Candra, K. Vetter, and A. Zakhor. Sensor fusion for semantic segmentation of urban scenes. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015.

[31] C. G. Snoek, M. Worring, and A. W. Smeulders. Early versus late fusion in semantic video analysis. In *Proceedings of the 13th annual ACM international conference on Multimedia*. ACM, 2005.

[32] T. W. Lewis and D. M. Powers. Sensor fusion weighting measures in audio-visual speech recognition. In *Proceedings of the 27th Australasian conference on Computer science-Volume 26*. Australian Computer Society, Inc., 2004.

[33] W. Luo, Y. Li, R. Urtasun, and R. Zemel. Understanding the effective receptive field in deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2016.

[34] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

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

[36] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[37] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.