Improving Movement Prediction of Traffic Actors using Off-road Loss and Bias Mitigation

Matthew Niedoba, Henggang Cui, Kevin Luo,
Darshan Hegde, Fang-Chieh Chou, Nemanja Djuric
Uber Advanced Technologies Group
{mniedoba, hcui2, kevin.luo, darshan.hegde, fchou, ndjuric}@uber.com

Abstract

There is a significant amount of recent literature in the domain of self-driving vehicles (SDVs), with researchers focusing on various components of the SDV system that have the potential to improve on-street performance. This includes work exploring improved perception of the SDV surroundings, or proposing algorithms providing better short-term prediction of nearby traffic actor behavior. However, in most cases, the authors report only aggregate metrics computed on the entire data, and often do not fully consider the bias inherent in the traffic data sets. We argue that this practice may not give a full picture of the actual performance of the prediction model, and in fact, may mask some of its problem areas (e.g., handling turns). We analyze the amount of bias present in traffic data and explore the ways to address this issue. In particular, we propose to use a novel off-road loss and standard bias mitigation techniques that result in improved performance. We further propose to avoid aggregate metrics and instead analyze performance on relevant subsets of the data, thus better capturing actual model capabilities. Moreover, we propose to measure a novel off-road error to complement commonly used prediction metrics. Extensive analysis of real-world data suggests benefits of the proposed approach for improving the performance of SDV technology.

1 Introduction

We are witnessing an unprecedented interest in the self-driving technology, with a large number of industry and academic researchers turning their attention towards this challenging task. This is evidenced by a substantial amount of recently published work at the top machine learning and robotics conferences on various areas of the problem, ranging from improvements to the perception systems used to understand the state of the world surrounding the self-driving vehicle (SDV) [26, 37], over new prediction methods that infer a short-term future of relevant traffic actors [5, 13], to motion planning approaches that help plan and optimize the SDV's path in this complex environment [8, 15]. Moreover, recent workshops focusing on the problem of SDVs at venues such as CVPR, ICRA, and NeurIPS further reflects significant interest and progress made in the field of autonomous driving [31], only expected to increase in the near future [18].

In this work, we focus on the problem of predicting the movement of traffic actors found in the SDV's surroundings [32], a critical part of autonomous technology. In particular, once the SDV manages to successfully detect and track nearby traffic actors, it also needs to understand how they will move in the short-term in order for both actors and the SDV to be safe during operation [13]. This is a very complex problem, as the inference model needs to take into account various inputs coming from the detection system, as well as other sources such as a 3-D map of the operating area [24] and interactions between the actors and the environment [33]. Researchers have proposed a number of

Machine Learning for Autonomous Driving Workshop at the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

methods to tackle this issue, ranging from tracking-based methods [40], methods using imitation or reinforcement learning [4], and end-to-end approaches [5, 27], to name a few.

While existing studies include extensive experiments showing the benefits of the proposed approaches, we argue that the presented offline evaluations may not provide the full picture, as researchers often consider only aggregate metrics and report performance results of their models on all available data. This practice may be suboptimal, as the traffic data is known to be highly imbalanced with majority of traffic actors simply driving straight [28]. As a consequence, aggregate metrics are dominated by the most frequent maneuvers, where the prediction models often perform well and are potentially not an ideal proxy for measuring on-road SDV performance. For example, left- and right-turning maneuvers in intersections could be of greater interest to practitioners, where accurate predictions are even more relevant for safe operations of the SDV.

To account for this issue we consider the recently introduced state-of-the-art RasterNet method [13], and experiment with traditional methods to reduce the effect of data imbalance using upweighting of under-represented categories. In addition, we introduce a novel *off-road loss*, that results in further performance gain over the baseline. We also propose to measure and report a novel *off-road error* to better capture model performance, in addition to commonly reported displacement, along-track, and cross-track error metrics. Lastly, we show that prediction metrics sliced by the maneuver mode paints a more detailed picture of the model performance, as opposed to only reporting aggregate metrics. The contributions of our work are summarized below:

- We discuss the problem of data imbalance in existing traffic data sets;
- We argue that reporting metrics sliced by maneuvers is critical for fully understanding the performance of prediction models;
- We propose to report a novel off-road error to complement commonly used prediction performance metrics used in the literature;
- We experiment with bias mitigation techniques and a novel off-road loss in the context of the state-of-the-art RasterNet method, and show improved aggregate and sliced metrics.

2 Related work

2.1 Deep learning for autonomous driving and data imbalance

Motion trajectory prediction of other traffic actors on the road is a critical component in many autonomous driving systems [4, 10, 40], as the SDV needs accurate trajectory predictions to safely plan the path through the dynamic traffic environment. The current state-of-the-art prediction work heavily depends on deep neural networks and large traffic data sets. For example, authors in [20, 22, 23, 25] used recurrent neural networks (RNNs) with Long Short-Term Memory (LSTM) or gated recurrent unit (GRU) to predict actors' future trajectories from past observed positions. Some researchers proposed to add additional scene context into their prediction models by rasterizing the map and actor's surroundings in a bird's-eye view (BEV) image and providing it as an input to convolutional neural networks (CNNs) [4, 9, 11, 13]. To explicitly model interactions between the traffic actors, authors of [2, 17, 29, 33, 38] modeled movement of all actors in the scene using social layers. Recent work by [17, 22, 29, 39] used GAN models to model the uncertainty and multimodality of actor's future motion. In [11] the authors tackled the multimodal prediction problem by having the model predict for each actor a small fixed number of trajectories along with their probabilities, and learning them jointly with a specially designed loss function.

Deep models require large amount of traffic data to train. However, recent work in traffic domain has shown that collecting data in real-world scenarios creates a natural data imbalance due to the long tail of driving behavior [3, 12, 36], which may impact the model performance in under-represented situations. The authors of [5] obtained a data set which was labelled according to 8 different high-level actions, and observed that actions like driving straight make up a vast majority of the trajectories. Several other works [11, 14, 28] noted similar imbalance between various vehicle maneuvers.

Countering the trajectory imbalance in self-driving has generally taken the form of downsampling majority classes [5] or weighting the loss differently on a per-example basis [36]. These schemes often rely on human-annotated scenario labels and are thus not scalable to varied scenario categories, granular categorization, and larger data sets.

2.2 Mitigations for unrealistic off-road trajectory predictions

Recently proposed ChauffeurNet [4] used an on-road loss to encourage the predictions to stay inside the road space, however our work differs from this method in several aspects. First, ChauffeurNet's on-road loss requires the trajectories to be predicted in the form of occupancy heatmaps. However, predicting such occupancy heatmaps is computational expensive, and most of the state-of-the-art trajectory prediction works (e.g., [6, 11, 29, 39]) do not predict occupancy heatmaps. Instead, common approach is to predict x- and y-coordinates of trajectory points, making the ChauffeurNet's on-road loss not directly applicable. The off-road loss proposed in this work, on the other hand, works on trajectory point coordinates and can be easily applied to the models discussed above. Second, ChauffeurNet treats the entire road surface as the drivable region (i.e., their ground-truth target road mask). However, as recent work [19] pointed out, the road surface alone does not solely describe an area where a vehicle can drive, and areas such as lanes in the opposite direction should also be marked as non-drivable. We also observe that incorrectly predicting an actor to unrealistically drive into an opposite-direction lane can cause even bigger problems for the SDV (e.g., causing the SDV to perform dangerous evasion maneuvers) than predicting actors to drive outside the road. In this work, we leverage the lane connectivity graph in the map to identify a list of lanes that each actor can realistically drive on, and include only these lanes in our drivable region set.

3 Methodology

In this section we first introduce the baseline RasterNet [9, 13], a recently proposed state-of-the-art approach for short-term prediction of actor movement. Then, we discuss potential concerns originating from biased data sets used for training, and propose approaches to mitigate these issues. It is important to emphasize that while we use RasterNet as a proof-of-concept in this work, the experiments and conclusions regarding mitigation of bias and off-road predictions are far more general, as we observed similar problems with other prediction approaches as well.

3.1 Description of the RasterNet model

The method assumes there exists a tracking system onboard a SDV, ingesting sensor data and allowing detection and tracking of traffic actors in real-time (e.g., using KF taking lidar, radar, and/or camera data as inputs). State estimate contains information describing an actor at fixed time intervals, including bounding box, position, velocity, acceleration, heading, and turning rate. Moreover, we also assume access to detailed map data of an operating area of the SDV, comprising road locations, lane and crosswalk boundaries, and other relevant map info. The resulting tracking and map data is then used as an input to the considered deep system.

We denote overall map data by $\mathcal{M}$, and a set of discrete times at which tracker outputs state estimates as $\mathcal{T} = \{t_1, t_2, \ldots, t_T\}$, where time gap Δt between consecutive time steps is constant (e.g., $\Delta t = 0.1s$ for tracker running at frequency of $10Hz$). State output of a tracker for the i-th actor at time t_j is denoted as $\mathbf{s}_{ij}$, where $i = 1, \ldots, N_j$ with N_j being a number of tracked actors at time t_j. Note that in general the actor counts vary for different time steps as new actors appear in and old disappear from the sensor range. Then, given data $\mathcal{M}$ and all actors' state estimates up to and including t_j (denoted by $\mathcal{S}_j$), the task is to predict sequence of future states (or trajectory) $\boldsymbol{\tau}_{ij} = [\mathbf{s}_{i(j+1)}, \ldots, \mathbf{s}_{i(j+H)}]$, where $\mathbf{s}_{i(j+h)}$ denotes state of the i-th actor at time t_{j+h}, and H denotes the number of future consecutive time steps for which we predict states (or *prediction horizon*). Past and future states are represented in an actor-centric coordinate system derived from actor's state at time t_j, where forward direction defines x-axis, left-hand direction defines y-axis, and actor's bounding box center defines the origin. We denote heading of an i-th actor at time t_{j+h} as $\phi_{i(j+h)}$, computed as an angle relative to the x-axis (i.e., forward direction at time t_j).

Let us assume a predictive model with parameter set $\boldsymbol{\theta}$, taking map $\mathcal{M}$ and state info $\mathcal{S}_j$ as inputs at time t_j, and outputting . The inputs are encoded as an overhead raster to represent surrounding context for the i-th actor (see Figure 2a for an example), used to predict trajectory of length H. We denote model outputs using the hat notation $\hat{\bullet}$, and for simplicity do not explicitly specify $(\boldsymbol{\theta}, \mathcal{M}, \mathcal{S}_j)$ as input arguments. Then, we write overall loss for the i-th actor at time t_j as the Euclidean distance between predicted and observed positions,

$$L_{ij} = \sqrt{(x_{i(j+h)} - \hat{x}_{i(j+h)})^2 + (y_{i(j+h)} - \hat{y}_{i(j+h)})^2}. \quad (1)$$

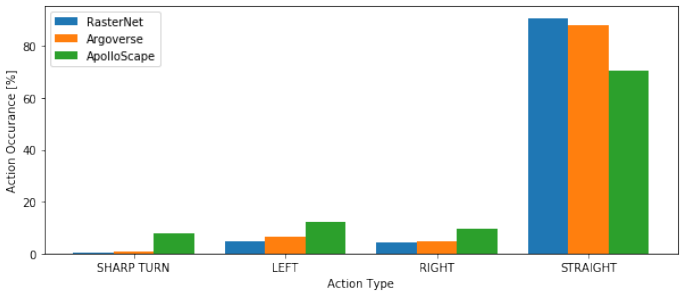

Figure 1: Maneuver frequency computed across three trajectory prediction data sets

As described in [11], this formulation can be extended to multimodal predictions. In particular, instead of predicting one trajectory, the model outputs M modes and their associated probability p_{ijm} modeled by a soft-max, with $m \in \{1, \ldots, M\}$. At each step during training we find a predicted mode closest to the ground-truth trajectory, indexed by m^*. Then, the final loss is defined as

$$\mathcal{L}_{ij} = \sum_{m=1}^{M} I_{m=m^*}(L_{ijm} - \alpha \log p_{ijm}), \tag{2}$$

where I_c is a binary indicator function equal to 1 if the condition c is true and 0 otherwise, L_{ijm} is computed as (1) taking only the m-th mode into account, and α is a hyper-parameter used to trade-off between the two losses (i.e., a log-likelihood of the observed data given a winning mode on one side, and a mode selection cross-entropy loss on the other). Note that, according to (2), during training we update the position outputs only for the winning mode and the probability outputs for all modes.

We can optimize equation (2) over all actors and all times that are available in the training data to learn the optimal model parameters,

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathcal{L} = \arg\min_{\boldsymbol{\theta}} \sum_{j=1}^{T} \sum_{i=1}^{N_j} \mathcal{L}_{ij}. \tag{3}$$

3.2 Exploring and mitigating bias in traffic data

In this section we present our analysis of several available traffic data sets, showing that the issue of bias is commonly encountered in practice. In particular, we analyzed the proprietary data used in the RasterNet paper [13], along with two trajectory prediction data sets recently made publicly available, the Argoverse motion forecasting data [7] and the ApolloScape trajectory data [35]. To categorize the actor trajectories into different maneuvers, we introduce several heuristic filters. For the purposes of this paper we consider *straight*, *left turn*, *right turn*, and *sharp turn* trajectories classes, where we define class filters by consider actor heading along the trajectory points.

Assume we are taking trajectory $\boldsymbol{\tau}_{ij}$ of length H as an input. Then, the class filters are defined by considering headings of trajectory points as defined below,

- going-straight trajectory filter:

$$I_{str}(\boldsymbol{\tau}_{ij}) = \begin{cases} 1, & \text{if } |\phi_{i(j+H)} - \phi_{ij}| \leq thresh_{str}, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

- left-turning trajectory filter:

$$I_{left}(\boldsymbol{\tau}_{ij}) = \begin{cases} 1, & \text{if } thresh_{str} < \phi_{i(j+H)} - \phi_{ij} \leq thresh_{turn}, \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

- right-turning trajectory filter:

$$I_{right}(\boldsymbol{\tau}_{ij}) = \begin{cases} 1, & \text{if } thresh_{str} < \phi_{ij} - \phi_{i(j+H)} \leq thresh_{turn}, \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

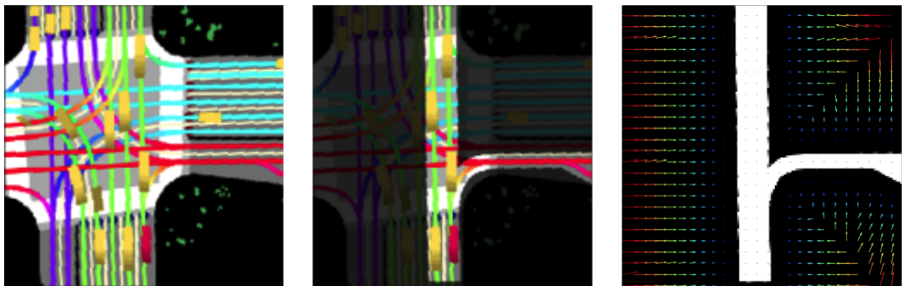

Figure 2: Visualization of the drivable region for a given actor: (a) RasterNet input; (b) drivable region highlighted in the raster image; (c) representation of the nearest drivable point, each pixel in the black region is associated with a white drivable point, where vectors indicate the direction and scaled distance to the nearest drivable point

- sharp-turn trajectory filter:

$$I_{uturn}(\boldsymbol{\tau}_{ij}) = \begin{cases} 1, & \text{if } |\phi_{i(j+H)} - \phi_{ij}| > thresh_{turn}, \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

We categorized the trajectories from the considered data sets using the filters defined in such a way, with the resulting data distribution shown in Figure 1 (we set the threshold parameters to $thresh_{str} = 20^\circ$ and $thresh_{turn} = 135^\circ$). We can see that in all three data sets more than two-thirds of trajectories belong to the straight class, with left-, right-, and sharp-turns occurring at significantly lower rates. It is clear that such bias is inherent to the traffic data found across the autonomous industry, where most of the time we observe common, straight trajectories, with much smaller fraction of the data representing more interesting, rare maneuvers.

As we show in Section 4, this inherent bias found in traffic data leads to the models performing substantially worse on the under-represented classes, which may have negative impact on SDV's on-street performance. To mitigate this issue we considered common strategies when dealing with imbalanced data sets, such as upweighting of rare classes obtained using the above filters. The results will be shown and discussed in the experimental section. Going beyond these approaches, in the next section we introduce a method that further improves performance of the prediction models when it comes to predicting under-represented traffic situations.

3.3 Penalizing off-road trajectories

When examining failure modes of trajectory predictions of a model trained using loss (3), we observed that large displacement errors often correspond to trajectories predicted to leave the region of the road that would be deemed safe to drive by a human driver. Common failures see actors encroaching into the oncoming traffic, driving onto the sidewalk, or exhibiting poor lane-following behavior. These are illustrated in Figure 3 in the experimental section, where we can see that the model may output suboptimal prediction in under-represented cases such as roundabouts or intersection types not seen in the training data. This behavior is especially noticeable in turning trajectories, where the model performs significantly worse than in other case. To address this issue, we propose two additional losses to (3) which penalize predicting driving that leaves the drivable regions.

As discussed previously, we assume we have access to detailed map data $\mathcal{M}$. To introduce the new losses, we are interested in the map data containing a directed graph that represents the lane structure of the roadway, as well as polygons that represent the physical extent of each lane. Given this data, for the i-th actor at time t_j, we generate a set of n 2-D lane polygons $\Pi_{ij} = \{p_1, p_2, ..., p_n\}$ which represent the BEV representation of the lanes which are reachable by traversing the directed lane graph from the actor's current position.

Given the drivable polygons, we generate a rasterized image representing the drivable surface formed from the union of polygons in Π_{ij} where each drivable pixel has a value of 1 and each non-drivable pixel has a value of 0 (see Figures 2a and 2b). Using this information, we generate an additional two channel raster image in which the value of each pixel corresponds to the coordinates of the nearest pixel which lies inside the drivable region (as illustrated in Figure 2c). We refer to the channels of

this raster image storing x- and y-coordinates of the nearest on-road point as $u(x, y)$ and $v(x, y)$, respectively. Given a predicted trajectory of length H, we introduce an additional off-road loss term,

$$\mathcal{L}_{ij}^{offroad1} = \frac{\lambda}{H} \sum_{h=1}^{H} \sqrt{\left(\hat{x}_{i(j+h)} - u(\hat{x}_{i(j+h)}, \hat{y}_{i(j+h)})\right)^2 + \left(\hat{y}_{i(j+h)} - v(\hat{x}_{i(j+h)}, \hat{y}_{i(j+h)})\right)^2}, \quad (8)$$

defined as the mean Euclidian distance between each predicted waypoint and its nearest drivable point as given by u and v, where λ is a hyper-parameter which controls the importance given to this off-road loss term.

In addition to the off-road loss approach (8), we also propose an upweighting scheme to reduce number of trajectories which deviate from the drivable region. Given u and v described previously, we penalize off-road false positives (i.e., trajectories predicted to go off-road for which the corresponding ground-truth waypoints remained inside the drivable region) as follows,

$$\mathcal{L}_{ij}^{offroad2} = \sum_{h=1}^{H} \alpha(\hat{x}_{i(j+h)}, \hat{y}_{i(j+h)}) \sqrt{(\hat{x}_{i(j+h)} - x_{i(j+h)})^2 + (\hat{y}_{i(j+h)} - y_{i(j+h)})^2}. \quad (9)$$

The upweighting is applied on a per-waypoint basis on the displacement error between the predicted and ground-truth waypoints, where α is a function which determines which waypoints to upweight

$$\alpha(x, y) = \begin{cases} \beta & \text{if } x \neq u(x, y) \text{ or } y \neq v(x, y), \\ 0 & \text{otherwise,} \end{cases} \quad (10)$$

where β is the upweighting factor controlling how much to upweight off-road false positive waypoints. Both losses (8) and (9) can be added directly to the loss (3) in order to enforce a higher penalty for off-road trajectory errors.

4 Experiments

4.1 Experimental setting

We collected 240 hours of data by manually driving SDVs in various traffic conditions (e.g., varying times of day, days of the week). The data contains significantly different number of examples for various actor types, namely 7.8 million vehicles, 2.4 million pedestrians, and 520 thousand bicyclists. Traffic actors were tracked using Unscented Kalman filter (UKF) [34], taking raw sensor data from the camera, lidar, and radar, and outputting state estimates for each object at $10Hz$. The filter is a default tracker on our fleet, trained on a large amount of labeled data, and tested on millions on miles (unfortunately, no other details can be given due to confidentiality concerns). We considered prediction horizon of $6s$ (i.e., $H = 60$) for vehicle actors, and used the same default rasterization scheme from [13] (with pixel size of $0.2m$). We implemented models in TensorFlow [1] and trained on 16 Nvidia Titan X GPU cards. We used open-source distributed framework Horovod [30] for training, completing in around 24 hours. We used a per-GPU batch size of 64 and Adam optimizer [21], setting initial learning rate to 10^{-4} further decreased by a factor of 0.9 every $20{,}000$ iterations. Models were trained end-to-end from scratch.

4.2 Metrics

Trajectory prediction performance is commonly evaluated using mean displacement error (MDE) and final displacement error (FDE), which we also consider in our experiments. In particular, we report ℓ_2 error at 3s (roughly corresponding to MDE) and at the final horizon of 6s (corresponding to FDE). We also report along-track (AT) and cross-track (CT) metrics [16], relevant for quantifying prediction system in the context of autonomous driving.

However, additional metrics are needed in order to quantify the frequency and magnitude of predictions which deviate from drivable regions. The reason is that these commonly used metrics fail to capture the importance of the environmental context when assessing the quality of trajectory predictions. To remedy this, we propose a novel metric to evaluate model performance by considering the predicted and ground-truth trajectories with respect to the contextual information from the actor's environment. In particular, we assume access to the same set of polygons Π_{ij} used in the off-road

Table 1: Comparison of prediction errors at 3s horizon sliced by maneuver types

	Overall				**Straight**				**Turning**			
Model	ℓ_2	CT	AT	OD	ℓ_2	CT	AT	OD	ℓ_2	CT	AT	OD
RasterNet	**1.52**	**0.34**	**1.39**	0.05	**1.46**	**0.28**	**1.37**	0.04	2.15	0.85	1.69	0.10
+ action upweight	**1.52**	**0.34**	**1.39**	0.05	1.47	0.29	**1.37**	0.04	**2.06**	**0.82**	**1.64**	0.11
+ off-road loss	1.58	0.37	1.44	**0.04**	1.52	0.32	1.41	**0.03**	2.13	0.84	1.68	**0.08**

Table 2: Comparison of prediction errors at 6s horizon sliced by maneuver types

	Overall				**Straight**				**Turning**			
Model	ℓ_2	CT	AT	OD	ℓ_2	CT	AT	OD	ℓ_2	CT	AT	OD
RasterNet	**4.79**	**0.90**	4.42	**0.14**	**4.54**	**0.61**	**4.37**	**0.11**	7.50	3.12	5.64	0.39
+ action upweight	**4.79**	0.93	**4.40**	0.15	4.59	0.67	4.39	**0.11**	**6.97**	2.90	**5.29**	0.44
+ off-road loss	4.91	0.96	4.52	**0.14**	4.72	0.73	4.49	**0.11**	7.03	**2.75**	5.47	**0.35**

losses to represent the drivable region. Then, given a set of drivable road polygons and a predicted trajectory, we define off-road distance (OD) for a given predicted position as

$$d^{offroad}_{i(j+h)} = \min_{p \in \Pi_{ij}} \ell_2\big((\hat{x}_{i(j+h)}, \hat{y}_{i(j+h)}), p\big), \tag{11}$$

where ℓ_2 is a Euclidean distance between predicted trajectory point $(\hat{x}_{i(j+h)}, \hat{y}_{i(j+h)})$ and polygon p.

In addition, we also use filters defined in Section 3.2 to slice the test data and provide detailed per-category errors, in addition to aggregate errors as commonly reported. As we will see, such evaluation provides important insights into the model accuracy on under-represented cases, painting a much clearer picture of the overall performance.

4.3 Results

In this section we present performance results of the competing approaches, where we report ℓ_2, AT, CT, and OD metrics, introduced in the previous section. To measure the performance of each multimodal method, we considered only the highest confidence trajectory for each test example.

To evaluate impact of the data imbalance discussed in Section 3.2, as well as the effectiveness of our proposed methods, we compared the following multimodal models:

- RasterNet: baseline model described in Section 3.1;
- action upweighting: an extension of RasterNet where left-, right-, and sharp-turn examples found using filters introduced in 3.2 were upweighted by a factor of 2;
- off-road loss: a model employing action upweighting, along with both off-road losses introduced in equations (8) and (9) (hyper-parameters were set to $\lambda = 0.25$ and $\beta = 5$).

The error metrics computed at the 3s and 6s-horizons are shown in Tables 1 and 2, respectively. We report metrics aggregated over all test examples (*Overall*), as well as metrics computed only on two maneuver types (*Straight* and *Turning*).

Let us first consider the RasterNet results, where we can see that the AT and CT errors differ significantly. CT metric is lower, however it is important to emphasize that cross-track errors may have a higher impact on the SDV's on-street performance. The reason is that errors in the cross-track direction may make autonomous vehicles very sensitive to incorrect predictions coming from vehicles located in neighboring lanes, unlike errors in the along-track direction where the SDV performance is usually much more robust. Having said this, it is interesting to consider results sliced by straight and turning maneuvers. As discussed in Section 3.2, straight trajectories cover vast majority of the traffic data, and the model metrics are expectedly better in these cases. However, the metrics are significantly worse on turning trajectories, which are often the cases that we mostly care about as these are the situations where good SDV performance is the most critical (e.g., behavior in interactions, interaction with turning actors, or vehicles cutting off the SDV). Thus, improving performance in such cases is of particular importance for the autonomous system.

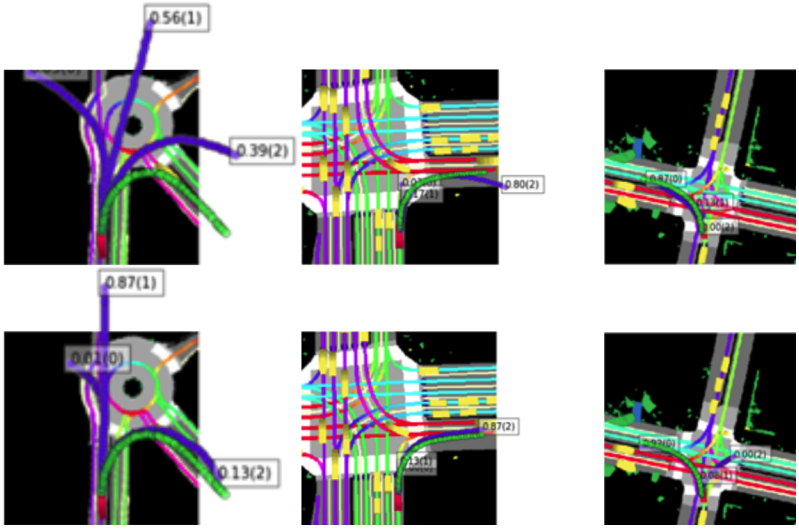

Figure 3: Qualitative examples comparing control (top) and off-road upweighting model (bottom), where green trajectory represents ground truth and blue trajectories represent model predictions (with mode probabilities given in text boxes); we can see that the model accounting for off-road errors exhibits significantly improved performance in these under-represented cases

Next we discuss an impact of using action upweighting to improve the RasterNet model. We can see that in aggregate the model has the same overall ℓ_2 error at both 3 and 6 seconds. Examining the sliced metrics, the action-upweighted model improved over RasterNet in turning scenarios at both horizons, but regressed slightly when it came to straight trajectories. This behavior was expected, as the upweighting of turning scenarios forces the model to give less priority to straight examples compared to RasterNet. Nevertheless, along the lines of our prior discussion, the improvement in the under-represented cases often leads to improved overall on-street performance of the SDV.

Going forward, we can see that the off-road model underperformed when examining the displacement, along-track, and cross-track errors at both 3s and 6s horizons. However, slicing by action category shows that this performance regression primarily stems from straight trajectories. On turning trajectories, the model improved over RasterNet in ℓ_2, AT, and CT errors at both 3s and 6s, and has the best cross-track error of any method at 6s. We hypothesize that this improvement in final cross-track error may be due to increased utilization of the map information and better understanding of potential drivable regions for the predicted trajectories.

Considering the off-road error metric, the off-road-loss model matched or improved over RasterNet and the action-upweight model across all horizons and slices. Improvements are most apparent on turning trajectories, which saw a 20% improvement over RasterNet at 3s and a 10% improvement at 6s. To further explore benefit of these improvements, we run qualitative analysis using several interesting examples, given in Figure 3. In particular, we considered left- and right-turning actor, as well as an actor approaching a roundabout. The first row shows output of the baseline RasterNet, with the output of the off-road-loss model illustrated in the second row. While the baseline predicted suboptimal trajectories that leave the actor's drivable regions, we can see that the improved models tends to follow lanes much better. In general, the trajectories do not cut corners, do not enter sidewalks, and overall have better lane keeping behavior. It is particularly interesting to see this improvement in the roundabout case, featuring very rare road geometry that is handled very well by the final model.

5 Conclusion

We discussed the common problem of imbalance in traffic data, shown to be inherent in a number of publicly available data sets. We considered recently proposed state-of-the-art model RasterNet, and through analysis of prediction metrics sliced by common vehicle maneuvers showed that the model is affected by this issue. We experimented with the traditional bias mitigation techniques that help improve the performance, and also proposed a novel off-road loss to gain further boosts in the model performance, as well as novel off-road metric to complement commonly used trajectory prediction metrics. Experiment on real-world data clearly show benefits of the proposed approaches.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015.

[2] 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*, pages 961–971, 2016.

[3] S. Atev, G. Miller, and N. P. Papanikolopoulos. Clustering of vehicle trajectories. *IEEE transactions on intelligent transportation systems*, 11(3):647–657, 2010.

[4] M. Bansal, A. Krizhevsky, and A. Ogale. Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. *arXiv preprint arXiv:1812.03079*, 2018.

[5] S. Casas, W. Luo, and R. Urtasun. Intentnet: Learning to predict intention from raw sensor data. In *Conference on Robot Learning (CoRL)*, 2018.

[6] 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*, pages 8483–8492, 2019.

[7] M.-F. Chang, J. W. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan, and J. Hays. Argoverse: 3d tracking and forecasting with rich maps. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[8] Y. F. Chen, M. Everett, M. Liu, and J. P. How. Socially aware motion planning with deep reinforcement learning. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1343–1350. IEEE, 2017.

[9] 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 *Workshop on 'Machine Learning for Intelligent Transportation Systems' at Conference on Neural Information Processing Systems (MLITS)*, 2018.

[10] A. Cosgun, L. Ma, et al. Towards full automated drive in urban environments: A demonstration in gomentum station, california. In *IEEE Intelligent Vehicles Symposium*, pages 1811–1818, 2017.

[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 *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[12] N. Deo, A. Rangesh, and M. M. Trivedi. How would surround vehicles move? a unified framework for maneuver classification and motion prediction. *IEEE Transactions on Intelligent Vehicles*, 3(2):129–140, 2018.

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

[14] J. V. Dueholm, M. S. Kristoffersen, R. K. Satzoda, T. B. Moeslund, and M. M. Trivedi. Trajectories and maneuvers of surrounding vehicles with panoramic camera arrays. *IEEE Transactions on Intelligent Vehicles*, 1(2):203–214, 2016.

[15] W. Gao, D. Hsu, W. S. Lee, S. Shen, and K. Subramanian. Intention-net: Integrating planning and deep learning for goal-directed autonomous navigation. *arXiv preprint arXiv:1710.05627*, 2017.

[16] C. Gong and D. McNally. A methodology for automated trajectory prediction analysis. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2004.

[17] 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*, pages 2255–2264, 2018.

[18] P. Hancock, I. Nourbakhsh, and J. Stewart. On the future of transportation in an era of automated and autonomous vehicles. *Proceedings of the National Academy of Sciences*, 116(16):7684–7691, 2019.

[19] C. Hughes, S. Chandra, G. Sistu, J. Horgan, B. Deegan, S. Chennupati, and S. Yogamani. Drivespace: Towards context-aware drivable area detection. *Electronic Imaging*, 2019(15):42–1, 2019.

[20] B. Kim, C. M. Kang, S. H. Lee, H. Chae, J. Kim, C. C. Chung, and J. W. Choi. Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network. *arXiv preprint arXiv:1704.07049*, 2017.

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

[22] 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*, pages 336–345, 2017.

[23] J. Liang, L. Jiang, J. C. Niebles, A. G. Hauptmann, and L. Fei-Fei. Peeking into the future: Predicting future person activities and locations in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5725–5734, 2019.

[24] 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)*, pages 641–656, 2018.

[25] Y. Ma, X. Zhu, S. Zhang, R. Yang, W. Wang, and D. Manocha. Trafficpredict: Trajectory prediction for heterogeneous traffic-agents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6120–6127, 2019.

[26] 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*, pages 12677–12686, 2019.

[27] P. Ondrúška, J. Dequaire, D. Z. Wang, and I. Posner. End-to-end tracking and semantic segmentation using recurrent neural networks. *arXiv preprint arXiv:1604.05091*, 2016.

[28] D. J. Phillips, T. A. Wheeler, and M. J. Kochenderfer. Generalizable intention prediction of human drivers at intersections. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 1665–1670. IEEE, 2017.

[29] 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*, pages 1349–1358, 2019.

[30] A. Sergeev and M. D. Balso. Horovod: fast and easy distributed deep learning in tensorflow. *arXiv preprint arXiv:1802.05799*, 2018.

[31] D. Tokody, I. J. Mezei, and G. Schuster. An overview of autonomous intelligent vehicle systems. In *Vehicle and Automotive Engineering*, pages 287–307. Springer, 2017.

[32] C. Urmson et al. Self-driving cars and the urban challenge. *IEEE Intelligent Systems*, 23(2), 2008.

[33] A. Vemula, K. Muelling, and J. Oh. Social attention: Modeling attention in human crowds. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–7. IEEE, 2018.

[34] E. A. Wan and R. Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, pages 153–158. Ieee, 2000.

[35] P. Wang, X. Huang, X. Cheng, D. Zhou, Q. Geng, and R. Yang. The apolloscape open dataset for autonomous driving and its application. *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[36] H. Xu, Y. Gao, F. Yu, and T. Darrell. End-to-end learning of driving models from large-scale video datasets. *arXiv preprint arXiv:1612.01079*, 2016.

[37] B. Yang, M. Liang, and R. Urtasun. Hdnet: Exploiting hd maps for 3d object detection. In *Conference on Robot Learning*, volume 87, pages 146–155, 2018.

[38] P. Zhang, W. Ouyang, P. Zhang, J. Xue, and N. Zheng. Sr-lstm: State refinement for lstm towards pedestrian trajectory prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12085–12094, 2019.

[39] 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*, pages 12126–12134, 2019.

[40] J. Ziegler, P. Bender, M. Schreiber, et al. Making bertha drive—an autonomous journey on a historic route. *IEEE Intelligent Transportation Systems Magazine*, 6:8–20, 10 2015.