Naturalistic Driver Intention and Path Prediction Using Recurrent Neural Networks

Alex Zyner, *Member, IEEE*, Stewart Worrall, *Member, IEEE*, and Eduardo Nebot, *Fellow, IEEE*

***Abstract*— Understanding the intentions of drivers at intersections is a critical component for autonomous vehicles. Urban intersections that do not have traffic signals are a common epicenter of highly variable vehicle movement and interactions. We present a method for predicting driver intent at urban intersections through multi-modal trajectory prediction with uncertainty. Our method is based on recurrent neural networks combined with a mixture density network output layer. To consolidate the multi-modal nature of the output probability distribution, we introduce a clustering algorithm that extracts the set of possible paths that exist in the prediction output and ranks them according to probability. To verify the method's performance and generalizability, we present a real-world dataset that consists of over 23 000 vehicles traversing five different intersections, collected using a vehicle-mounted lidar-based tracking system. An array of metrics is used to demonstrate the performance of the model against several baselines.**

***Index Terms*— Trajectory prediction, intersection assistance, mixture density networks, recurrent neural network.**

I. Introduction

DRIVING vehicles is a highly skilled task that requires extensive understanding of the intentions of other road users. This knowledge allows drivers to safely navigate an area through other traffic. While this may become second nature to an experienced human driver, properly understanding the intentions of other drivers is still an unsolved problem for Advanced Driver Assistance Systems (ADAS), and by extension, autonomous vehicles. Moreover, this problem is studied extensively on highways with datasets such as NGSIM US 101 and I 80 [1], or on highly structured intersections with multiple lanes and traffic lights. There is little focus on smaller, neighborhood intersections that commonly do not have signals, or even proper road paint. Being able to safely navigate these highly dynamic scenarios is equally as critical for autonomous vehicles to function properly. A significant proportion of accidents occur at intersections, of which 84% can be attributed to either recognition error or decision error of a driver [2]. In this paper we propose a method for predicting a driver's intention with uncertainty, which produces a multi-modal output distribution over the predicted path a driver may take, as shown in Figure 1. This is important as considering uncertainty is critical for autonomous vehicles to make an informed decision, and the multi-modal nature of the output matches the multi-modal nature of intersections. In an intersection, there may be multiple paths to take, but the average of two solutions may not be a valid solution. Therefore proposing multiple paths a vehicle may take and ranking them allows for a better representation of a vehicle's predicted intention.

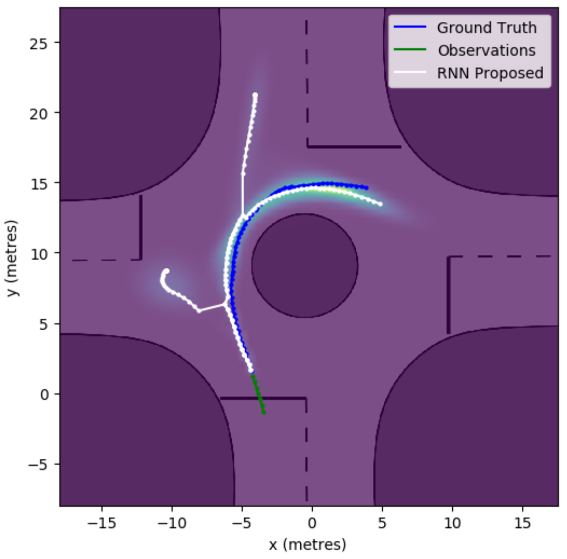


Fig. 1. Predictions of the next 5 seconds after a vehicle has entered an intersection. Here, each mode of the output is shown as a path in white, and the probabilities as a heatmap. The 0.56 seconds of observation data are shown in green, and the ground truth is in blue.

The model presented is a method for predicting the path of a driver at a roundabout. Using a dataset of several roundabouts, we are able to prove its generalizability across different, similarly shaped intersections. For full integration into an autonomous vehicle, there would need to be a differently trained network for each intersection type, to be used as a library. As the method is generalizable, each intersection does not have to be extensively surveyed. Roundabouts were chosen as the intersection case study as these intersections are not structured, and have no traffic lights denoting which vehicle has right of way. This makes these intersections highly dynamic, and so it is very difficult to predict a driver's path and intention at these intersections.

Manuscript received July 26, 2018; revised November 25, 2018 and February 3, 2019; accepted March 31, 2019. Date of publication May 13, 2019; date of current version March 27, 2020. This work was supported in part by the Australian Research Council Discovery (ACFR) under Grant DP160104081 and in part by the University of Michigan Ford Motors Company Contract Next Generation Vehicles. The Associate Editor for this paper was C. Guo. *(Corresponding author: Alex Zyner.)*

The authors are with Department of Engineering and Information Technologies, The University of Sydney, Sydney, NSW 2006, Australia (e-mail: a.zyner@acfr.usyd.edu.au; s.worrall@acfr.usyd.edu.au; e.nebot@acfr.usyd.edu.au).

Digital Object Identifier 10.1109/TITS.2019.2913166

The following contributions are presented in this paper:

Driving vehicles is a highly skilled task that requires extensive understanding of the intentions of other road users. This knowledge allows drivers to safely navigate an area through other traffic. While this may become second nature to an experienced human driver, properly understanding the intentions of other drivers is still an unsolved problem for Advanced Driver Assistance Systems (ADAS), and by extension, autonomous vehicles.

- We present a model that produces a multi-modal path prediction with uncertainty, and is clustered into a meaningful output.
- Real-world, real-time - Unlike many works that use the NGSIM dataset, we do not filter the input data at all, and so all evaluations taken at time t does not rely on data taken at time t_{++} for filtering or analysis. The data was collected with a lidar enabled vehicle as opposed to an overhead camera, and so it includes all data tracking noise, and it mimics the perspective issues of a vehicle driving through the scene.
- Generalizable across intersections - We demonstrate that this model generalizes between similarly sized intersections by using a dataset that spans multiple intersections of a similar type.
- Large, naturalistic dataset - We present a dataset that accumulates over 60 hours of vehicle data of naturalistic human drivers traversing five different roundabout style intersections in Sydney. To the author's knowledge, this is the largest dataset of unsignalized intersections publicly available, and can be downloaded at: http://its.acfr.usyd.edu.au/datasets/

The remainder of the paper is outlined as follows: Section II presents related work. The problem is formally defined in Section III, which includes describing the intersection, the frame of reference of the data, and the testing method. The proposed algorithm is described in Section IV, including the network style and output clustering. The experiments are described in Section V, including the dataset and how it was collected. Finally, the results and conclusions are presented in Sections VI and VII.

II. Related Work

There has been significant work in the area of predicting human driver intention, as it is critical for autonomous vehicles to co-exist on roads with human drivers. This problem can be formulated in multiple ways, and there are a number of different approaches taken in the literature. These can be loosely grouped into maneuver based models, path prediction models, and interaction aware models.

Maneuver based models are those that predict the intention of a driver by classifying intention into a known set of groups, which can include lane changing, stopping, and turns at an intersection. Once the maneuver has been predicted, the vehicle's trajectory can be assumed to match that of the particular maneuver. There have been several approaches to this problem recently, with the use of Support Vector Machines [3], Hidden Markov Models [4], and Bayesian Networks [5]. The author's previous work [6] demonstrates the classifying accuracy of Long Short-Term Memory (LSTM) given data from a lidar based smart vehicle at an unsignalized intersection. Phillips *et al.* [7] demonstrate the classification accuracy of LSTMs at multi-lane, signalized intersections with dedicated turning lanes, as in the NGSIM Lankershim and Peachtree datasets.

Path prediction models attempt to produce a future trajectory of the vehicle given some data on the vehicle's past. Often, physical based kinematic or dynamic models are used in conjunction with Switching Kalman Filters [8], Monte Carlo Simulations [9], or Variational Gaussian Mixture Models [10]. These methods will output a single prediction proposal, and give a measure of uncertainty. A maneuver recognition system can be combined with a Gaussian Process based path predictor to allow multi-modal prediction [11].

The previous methods only attempt to explain the movement of the target vehicle around traffic infrastructure, without any knowledge of the neighboring vehicles, or how they interact. Interaction aware models address this problem by incorporating movement, and sometimes prediction of the surrounding vehicles in the scene. Recently, there has been work in this area with a focus on vehicles on highways using processed visual data of surrounding vehicles [12], radar data [13] or simulated lidar returns of surround vehicles [14]. Models such as those presented by Lee *et al.* [15] or Schmerling *et al.* [16] that use a Conditional Variational Autoencoder will produce a prediction without uncertainty, but the model may be sampled many times to produce multiple predictions, which may be combined to estimate uncertainty. Real world data for this problem is sparse, as while it is easy to capture vehicles in a scene, it is very difficult to guarantee that the dataset encompasses all the vehicles that the target vehicle interacts with. Given the similarities of predicting drivers to predicting pedestrian movement, Lee *et al.* [15] demonstrate their results on vehicle trajectories taken from the KITTI Dataset, and pedestrian sequences taken from the Stanford Drone dataset [17], as it is an unobstructed overhead camera view.

Other works have instead chosen to model interactions through a simulator, by either simulating the entire interaction [18], or by using a driving simulator to capture the actions of human drivers [16]. Validating the use of simulators is difficult when modeling real-world scenarios that involve risk as there is no equivalent risk present in the simulator [19].

A commonly used dataset for this work is the NGSIM highway datasets, US-101 and I-80. This dataset was collected in 2005 using multiple overhead cameras observing sections of highway. This data was taken over three sections of 15 minutes each and contains trajectories of roughly 5000 vehicles. Visual tracking techniques were used to extract vehicles trajectories from the image data at a rate of 10Hz. Given the nature of tracking vehicles from a distant camera, the data suffers from considerable tracking noise [20], and even with aggressive filtering and correction techniques, significant problems still exist such as multiple collisions [21] that do not exist in the real-world sample. We overcome these limitations with a modern data collection vehicle, as later described in Section V-A.

Very recently there has been a focus on using deep learning to solve this problem, namely using recurrent neural networks (RNNs). This network style has been shown to be particularly useful for maneuver classification, with Phillips *et al.* [7] using the Lankershim and Peachtree datasets to predict the turn a driver will take at a multi lane intersection based on surrounding vehicles, lateral lane position, which lane a vehicle is in, and legal turns that can be made from said lane. A similar technique can also be applied to smaller single lane intersection without signals based on vehicle location and speed [22].

Given the recurrent nature of RNNs, they have been shown to work with sequence generation quite effectively, including handwriting, text generation [23], and freehand drawings [24]. The model is given a snippet of history that ends at time t, and then the model is trained to produce a predictive distribution over $t+1$. To generate longer sequences, a single sample from the prediction distribution is taken, and fed into the model again in a feed-forward fashion to generate a trajectory as long as desired.

This technique has been used to predict pedestrian intention, as this movement is very non-linear and is significantly influenced by the movement of surrounding agents. A model that incorporates a 'social-pooling' layer to explain this interaction is presented by Alahi *et al.* [25], which produces a path proposal with uncertainty. A similar technique is proposed by Pfeiffer *et al.* [26], however they only use the RNNs to encode the history information, and choose to use a fully connected layer to output a set of velocities that represent the predicted path. Recently this technique has been extended to predicting the trajectory of vehicles on US highway 101 and interstate 80 in the NGSIM dataset. Deo and Trivedi [27] use social pooling layers to encode data about surrounding vehicles on a highway, and predict various path proposals based on a maneuver classification.

While the highway problem is highly studied, there is less focus on intersections, which account for a significant proportion of accidents — around 40% [2]. Intersection datasets include the Ko-PER project, but this is taken from an overhead perspective to simulate smart infrastructure [28]. As previously mentioned, NGSIM dataset Lankershim and Peachtree are also often used, and this is data collected from an overhead camera passed through a tracking algorithm. For data that is taken onboard a vehicle, it is common to take data annotations from KITTI and use that in lieu of an actual tracking algorithm, further abstracting the solution away from a practicable pipeline [29]. Most of this data is of a structured intersection with easily visible lane markings, and traffic signals, and there is little focus on unstructured, unsignalized settings.

III. Problem Definition

While there has been a large focus for driver behavior in structured environments, namely those with strictly defined lanes and with traffic signals, there is significantly less focus on unsignalized road scenes. These urban areas commonly do not have road infrastructure to enforce strict behavior and ordering of vehicles, such as traffic lights or turning lanes. As such, there is a wide range of driving styles that a driver may exhibit when traversing intersections, which significantly increases the complexity of the problem. Given the non-linear trajectory behavior that exists at intersections, standard physical-model based tracking methods fail very quickly, and so we present a new method to solve this problem.

A. Data Definition

To study unsignalized urban intersections, we collected a real-world dataset of vehicles passing though several different roundabouts in Sydney, Australia as outlined later in Section V-A. The data consists of vehicle tracks, $\mathbf{V} = [\mathbf{V}_1, \mathbf{V}_2, \mathbf{V}_3, \ldots, \mathbf{V}_n]$ where each track in $\mathbf{V}$ contains the whole history of positions in lateral (x_t) and longitudinal (y_t) coordinates, as well as heading (θ_t) and velocity (v_t) that exist over many time-steps t. These tracks are then split into all possible snippets for a given observation length (h) and prediction length (p) using a sliding window technique. Our goal for this problem is to predict a probabilistic estimate $\hat{\mathbf{Y}}$ of the future path of the vehicle $\mathbf{Y}$ given a short observation sequence $\mathbf{X}$, where:

$$\begin{aligned}
\mathbf{X_t} &= [\mathbf{x}_{t-h+1}, \mathbf{x}_{t-h+2}, \mathbf{x}_{t-h+3}, \ldots, \mathbf{x}_t] \\
\mathbf{x_t} &= [x_t, y_t, v_t, \theta_t] \\
\mathbf{Y} &= [\mathbf{y}_{t+1}, \mathbf{y}_{t+2}, \mathbf{y}_{t+3}, \ldots, \mathbf{y}_p] \\
\mathbf{y_t} &= [x_t, y_t]
\end{aligned}$$

Here, $\mathbf{X}$ is the collection of all observations for a particular track snippet leading up to time t, x and y are absolute co-ordinates in meters with respect to the intersection, v is the vehicle's velocity, θ is the vehicle's orientation, $\mathbf{Y}$ is the collection of the future track after time t of length p and is the ground truth, and $\hat{\mathbf{Y}}$ is the probability estimate over $\mathbf{Y}$.

An interesting property of a roundabout is that the time a vehicle is present on a roundabout is influenced by the maneuver that vehicle is performing, that is to say left turns are significantly shorter in distance and time than right turns. This means that no one fixed prediction time horizon p can properly represent the path a vehicle may take for all turns. We implement a padding technique to overcome this, allowing the shorter tracks to be properly represented when the overall track length is shorter than $h + p$.

B. Frame of Reference

Each of the observed vehicles needs to be shifted into a common frame of reference for a prediction algorithm to function properly. The reference frame for this data is such that the origin lies at the center of the approach road, which is generally where a 'give way' sign is placed to mark the intersection. Each vehicle begins traveling upwards, from the bottom of the frame towards the origin. This alignment allows for a common reference for all observed vehicles, and remains in Cartesian coordinates. Frenet coordinates were not used as there are no clear lane markings, and the distinction between the entrance lane and the center, circular lane is ambiguous.

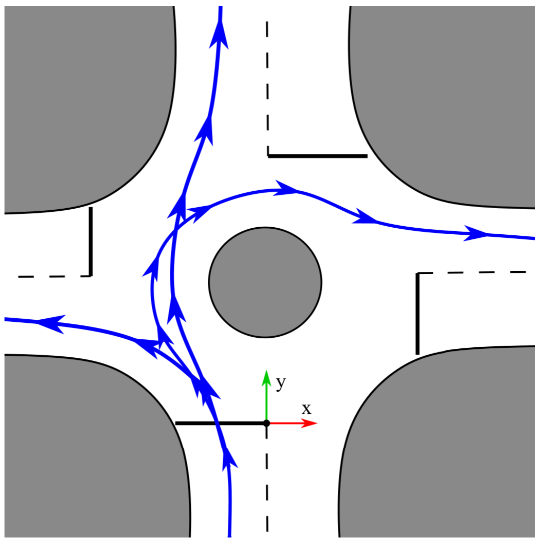


Fig. 2. A diagram of a typical single lane urban roundabout. Here, all the recordings are normalized such that each vehicle enters from the bottom of the diagram. The intersection entrance line is the black line the vehicle crosses when entering an intersection, in the lower half of this diagram. The origin of the coordinate frame is marked in the lower section of the diagram.

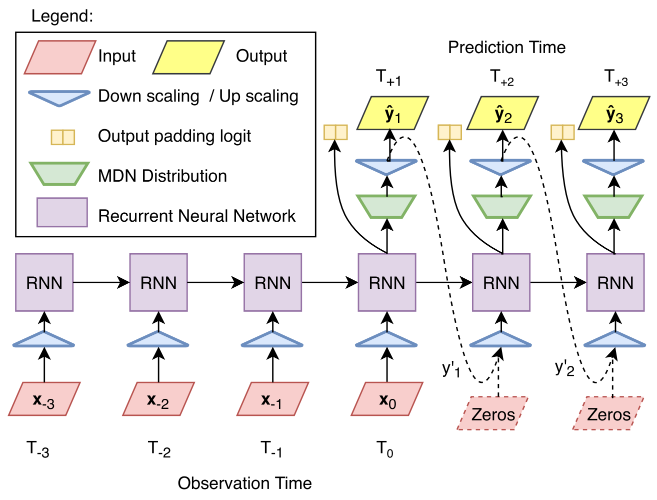


Fig. 3. The architecture of the Recurrent Neural Network that depicts how the system recurses forward, and the output layers for the mixture density network. Note that at each time-step the data is downsampled and upsampled such that the output of the network is in real-world units, in this case meters. Two data input methods may be used at prediction time. The method used for RNN-FF is such that single random sample is to be drawn from the posterior distribution and used as the input for the next time step, as shown as a dashed line. The second method, used in RNN-ZF is to simply feed zeros to the network instead of a single sample, which is shown as a dashed input block and denoted as y'_t. These two methods cannot co-exist. Each recurrance of the neural network is input with the data recorded at the corresponding timestep.

C. Testing

A full track of a vehicle consists of data during the approach of the vehicle to the roundabout, data when the vehicle is traversing the roundabout, and data after the vehicle has left the roundabout. If the chosen track snippet time t is after the vehicle has left the roundabout, predicting their intent has become trivial as their intent has already been demonstrated. If the snippet contains observations $\mathbf{X}$ that are well before the vehicle has approached the intersection, the driver of the observed vehicle may not have even decided which direction they will turn, which makes accurate intention prediction impossible. In this way, the problem becomes easier over time as there is a gradient of difficulty that exists from impossible to trivial. To rectify this, when the models are tested and scored only the track snippet where $\mathbf{x_t}$ lies at the intersection entrance is used. The intersection entrance line is the line of which, if the vehicle crosses it must commit to passing through the intersection. It is possible that vehicles have stopped before this line to give way to vehicles already on the roundabout, before entering the intersection. In this way the set of data fed into the network is data during the approach, and the model is used to predict the trajectory of the vehicle as it traverses the roundabout.

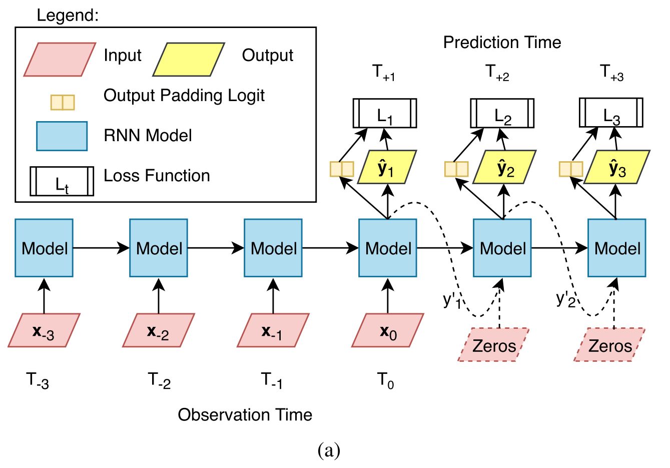
IV. Model

The proposed model is based around a Recurrent Neural Network (RNN). This is a style of neural network that has a copy of the network for each time-step of the input data, and these networks are chained together to form a sequence. This allows the network to be dynamic - it can take input, and produce output of arbitrary sequence lengths. This makes this type of network ideal for time-series data, as the sequential nature of the network matches that of the sequential nature of the input data. To solve the multi-modal nature of the data, a Mixture Density Network (MDN) is used, allowing for a probabilistic output. The loss function for this then becomes the probability that the ground truth could be sampled from the output probability density. The output mixtures of the network are then consolidated through a clustering technique.

Thus, the input of the network is $\mathbf{X_t}$, which is a track recording sample of length h, and at each time-step of the RNN the network is input with $\mathbf{x_t}$. This consists of co-ordinates in meters of the vehicle relative to the intersection reference point as well as heading in radians and speed in meters per second. The output of the network is a probability distribution estimate of the future path of the vehicle $\hat{Y}$, represented by a mixture of Gaussians for each time step $\hat{y}_t$, as described in the following section. The time-step used is 12.5Hz, and all hyper parameters are listed in section: V-B.

A. Architecture

The core architecture of the network presented in this paper is similar to the Recurrent Neural Network architecture found in the handwriting generation of Graves [23]. These networks are constructed of a recurrent neural network, a type of network that is suited for analysis of sequential data that is in discrete steps, such as logged sensor data. The network consists of a copy of the weights of the network for each timestep, where each of the recurrent portions of the cells may take input from the previous timestep's recurrent cell, and from the new observation data that is fed into the network. Similarly, the recurrent section of the network has two outputs: a hidden vector that is passed to the recurrent cell one time-step into the future, and a traditional output that is passed to further layers to finally give the model output. The architecture of this can be seen in Figure 3. This particular network style is described as an encoder-decoder network, with shared weights. The encoder



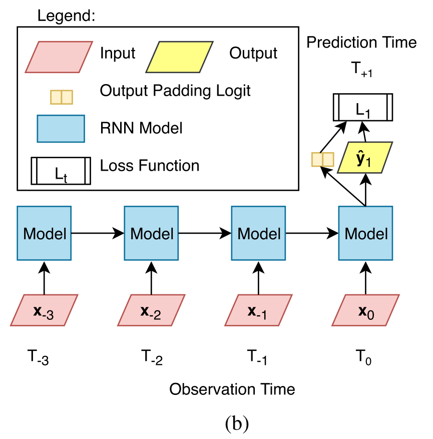


Fig. 4. A diagram depicting the two training styles compared in the results. The top Figure 4a depicts the loss generation for the **RNN-FF** and **RNN-ZF** models, where the whole output path is generated at training time and used in the loss function. The lower Figure 4b shows the training for the **RNN-FL** model, where only the first element of the trajectory is used in the loss function.

refers to the section of the RNN that exists between $t-h$ and t, the observation stage. The decoder refers to the section of the architecture that exists in prediction time; it is the part of the network that generates a path prediction.

Many of these models are trained using a history of past movement, and then used to predict only one time-step in advance, which is then compared to ground truth to generate a loss for training. This training style can be seen in Figure 4b. At inference time, a single sample is taken, which is then passed into a recurrence of the network at the next 'true sample'. This feedforward technique is used for generating a whole track, and is depicted in Figure 3. This is in contrast to the model presented in this paper, where a full prediction is generated at training time, and used for loss, as seen in Figure 4a. We compare three training techniques in the results section: using $t+1$ loss only (**RNN-FL**), sampling and feedforward for the whole prediction sequence (**RNN-FF**), and feeding zeros and still predicting the whole output sequence (**RNN-ZF**). These are explained in more detail in Section IV-D.

B. Model Output

1) Dynamic Data Scaling: For a neural network to function properly, the input data should be normalized. This is also true of most statistical techniques, and is usually done once during the data preprocessing step. For easier integration into real-world data, we have chosen to implement this as the very first network layer, with weights and biases that are fixed in the model as the normalization parameters. The final output is then scaled back to real-world units using the same parameters. These parameters are generated using the training data only, and saved with the network. Thus the input to the network is x_t and the output of the network is $\hat{y}_t$, where:

$$x_t = \frac{\mathbf{x}_t - \mu_\mathbf{x}}{\sigma_\mathbf{x}} \tag{1}$$

The values for $\mu_\mathbf{x}$ and $\sigma_\mathbf{x}$ are determined for each element in $\mathbf{x}$ in the training set, and then are fixed as the first layer of the network.

2) Sequence Length Padding: As mentioned in Section III-A, the output data is not all the same length, as vehicles turning left exit the scene much faster than those traveling straight or turning right, so naturally the sequence length is shorter. As such we cannot nominate the network to run for N number of steps and appropriate a meaningful output, as there may not be ground truth data to compare to. To alleviate this problem, we introduce a padding logit, to allow the network to nominate whether the vehicle has left the intersection, and the rest is padding data. The data used during padding is the last known position of the vehicle. This prevents the network from over-fitting later elements in the prediction sequence to those that only exist in longer sequences, i.e. right turns.

3) Mixture Density Network: Ideally we would like to predict the single most likely outcome, leading to a regression based approach. This does not work well as there are several distinct solutions, all of which may have some level of probability, but the average of these solutions is not another solution. So, a Mixture Density Network output layer can be used to allow a network to propose multiple solutions, and produce relative probability between them. This is achieved via a weighted mixture of Gaussian distributions, thereby creating a Gaussian Mixture Model.

The output, $\hat{y}$ of the network is then used to produce MDN parameters as follows:

$$\hat{y}_t = \left(\hat{p}_t, \{\hat{\pi}_t^j, \hat{\mu}_t^j, \hat{\sigma}_t^j, \hat{\rho}_t^j\}_{j=1}^M\right) \tag{2}$$

Here the mean μ_t and standard deviation σ_t are two dimensional vectors that exist over the parameters in $\mathbf{y_t} = [x_t, y_t]$ while the others are scalar, with p being the probability of this timestep being padding, π being the weight of each density in the mixture and ρ being the correlation coefficient. The following equations are used to enforce some constraints on the parameters of the MDN, as well as up-scaling the output such that the final results matches the input data for scale.

$$p_t = \frac{1}{1+\exp\left(\hat{p}_t\right)} \implies p_t \in (0,1) \tag{3}$$

$$\pi_t^j = \frac{\exp\left(\hat{\pi}_t^j\right)}{\sum_{j'=1}^M \exp\left(\hat{\pi}_t^{j'}\right)} \implies \pi_t^j \in (0,1), \quad \sum_j \pi_t^j = 1 \tag{4}$$

$$\mu_t^j = \sigma_s \hat{\mu}_t^j + \mu_s \implies \mu_t^j \in \mathbb{R} \tag{5}$$

$$\sigma_t^j = \sigma_s \exp\left(\hat{\sigma}_t^j\right) \implies \sigma_t^j > 0 \tag{6}$$

$$\rho_t^j = tanh(\hat{\rho}_t^j) \implies \rho_t^j \in (-1, 1) \tag{7}$$

The probability density $\Pr(x_{t+1}|y_t)$ of the next input x_{t+1} given the output vector y_t is defined as follows:

$$\Pr(x_{t+1}|y_t) = \sum_{j=1}^{M} \pi_t^j \; \mathcal{N}(x_{t+1}|\mu_t^j, \sigma_t^j, \rho_t^j) \tag{8}$$

where

$$\mathcal{N}(x|\mu, \sigma, \rho) = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}} \exp\left[\frac{-Z}{2(1-\rho^2)}\right] \tag{9}$$

with

$$Z = \frac{(x_1-\mu_1)^2}{\sigma_1^2} + \frac{(x_2-\mu_2)^2}{\sigma_2^2} - \frac{2\rho(x_1-\mu_1)(x_2-\mu_2)}{\sigma_1\sigma_2} \tag{10}$$

The number of mixtures M needs to be greater than the number of expected modes in the output, in this case 6 was used.

4) Loss: The loss for the MDN reconstruction is then the probability that the ground truth could be sampled from the output MDN distribution. A cross entropy loss is used for the padding output p, against the ground truth padding g. The final loss is then a combination of the padding cross entropy (CE) loss, and the negative log probability loss:

$$\mathcal{L}(\mathbf{x})_{CE} = -(g \log(p) + (1-g)\log(1-p)) \tag{11}$$

$$\mathcal{L}(\mathbf{x})_{MDN} = -\log\left(\sum_j \pi_t^j \mathcal{N}(x_{t+1}|\mu_t^j, \sigma_t^j, \rho_t^j) \times \begin{cases} 1 & \text{if } g = 0 \\ \beta & \text{otherwise} \end{cases}\right) \tag{12}$$

$$\mathcal{L}(\mathbf{x}) = \sum_{t=1}^{T} (\mathcal{L}(\mathbf{x})_{MDN} + \alpha\mathcal{L}(\mathbf{x})_{CE}) \tag{13}$$

where μ_s and σ_s are the mean and standard deviation of each dimension of the training data, and α and β are hyperparameters used to balance the two loss functions. The parameter β is non-zero to promote the network to nominate the last known predicted position of a vehicle before the vehicle exited the scene.

During feedforward sampling, the final two parameters in $\mathbf{x}$: v and θ can be determined via computing the magnitude and orientation of the vector between $\mathbf{x_t}$ and $\mathbf{x_{t-1}}$.

Given the strong multi-modal nature of the data, we found that the model started to ignore inputs during the feedforward stage. Once the output splits into multiple distinct possibilities, randomly sampling from the posterior produced a vehicle that effectively exists in two places at once, with an impossibly high velocity. As this phenomenon obviously isn't physically possible, it does not exist in the dataset, and so the model eventually ignores the feedforward input. To confirm the model does ignore the feedforward sample, we compare a single sample feedforward network with a network fed with only zeros during prediction time. This is further discussed in the results section VI-C.

Algorithm 1 MultiPAC

```
1: Ignore all mixes with $\pi < (\tau/n_{mixes})$
2: Run DBSCAN on mix centroids for each timestep
3: Declare nodes for each of the cluster outputs from DBSCAN
4: Construct a tree from these nodes
5: Declare the centroid of each node as the member's average in $x$ and $y$ weighted by $\pi$
6: Assign children to the closest parent based on euclidean distances of centroids
7: Return a list of all paths from leaf to root. These are the multiple modes of the model
```

C. Output Consolidation

The model by nature is a mixture of Gaussians, and these Gaussians often do not converge to the same solution, nor are they completely separate from each other. This is representative of the problem, as there are only a limited number of maneuvers possible at an intersection. Given the large number of output densities, there needs to be a clustering algorithm to consolidate the output and present it in a succinct and meaningful representation.

The presented clustering technique, the multiple prediction adaptive clustering algorithm (Multi-PAC), is presented in Algorithm 1. It groups all the possible outputs into paths, and ranks them according to their assigned probability. To do this, less meaningful mixes are first ignored, where their assigned weight is less than a threshold t. Then, DBSCAN [30] is run to group mixes together into several larger clusters. The centroids of these clusters are generated by finding the weighted average center of all the Gaussians in a group. These groups can then be assigned as nodes. Afterwards, a tree can be constructed where the depth of each node is fixed to the timestep of that node, and children are assigned to the closest parent. The final output of this algorithm is the path from each leaf to the root, which correlates to each mode of the multimodal output. All the paths are ranked in order of relative weight: $\sum_{path} \pi$. This prevents the algorithm from being fixed to only present a specific number of solutions.

D. Proposed Models

The following are the architectures tested:

- **RNN-FF** RNN Feed Forward - The model described in Section IV is presented, where a single sample is taken from the output distribution and used as input for the next time-step. This is also done at training time, and the whole ground truth is used to generate the loss, as seen in Figure 4a. As there is random sampling during training, the gradients do not propagate through the sampler, but they do propagate through the recurrent layers of the RNN.
- **RNN-ZF** RNN Zero Feed - The model described in Section IV is presented, where only zeros are used as input for the next time-step during path generation. This is also done at training time, and the whole ground truth is used to generate the loss, as per Figure 4a. The input

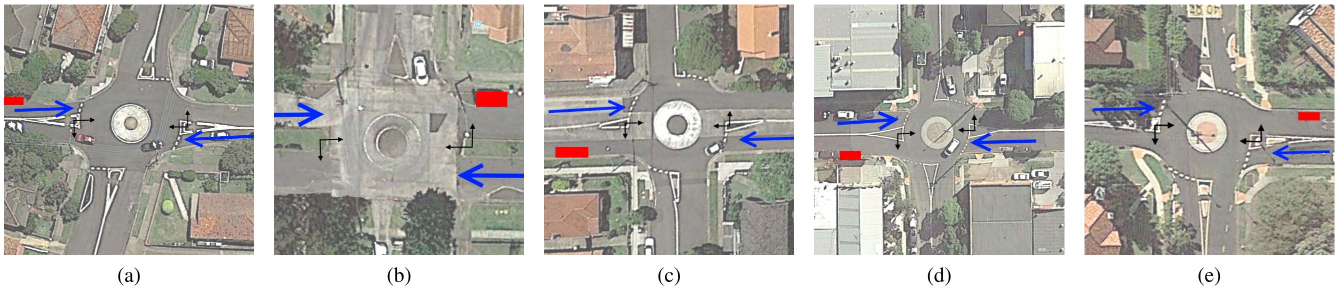


Fig. 5. Satellite views of each of the five roundabouts in the dataset: Queen-Hanks 5a, Leith-Croydon 5b, Roslyn-Crieff 5c, Orchard-Mitchell 5d, and 5e Oliver-Wyndora. Here only the two entrances that are most visible from the data collection vehicle are used, and these are shown via a blue arrow. The origin of the reference frame for each approach is depicted via black arrows. The red square is where the data collection vehicle was parked. Note that there is no lower exit for the Leith-Croydon intersection 5b, so vehicles may travel straight, or turn to use the exit at the upper side of the figure.

to the network cannot simply be turned off as the layer expects input data, so a dummy variable of zero was used for experimentation.

- **RNN-FL** RNN First Loss - The decoder is only run for one time-step, and such the loss is only generated for t_{+1}, as seen in Figure 4b. At run time a single random sample is drawn from the output distribution and used as the input for the next time-step. This is the training method used for previous MDN based sequence to sequence tasks in works such as handwriting prediction [23].

V. Experimental Setup

This section describes the methodology used for testing the algorithm, the data used for collection, and works used for comparisons.

A. Dataset Description

Roundabouts were chosen as the case study for this work. These small, unsignalized intersections involve highly variable maneuvers and are significantly more complex than the vehicle maneuvers that occur on highways and other structured environments. A dataset sufficient to study these aspects is not publicly available, and so collection of data was necessary. Note that the data used in this is an expansion of the data used in the author's previous work. It now includes multiple intersections that allow for proof of algorithm generalizability across similar styles of intersection. The data used in this work can be downloaded at: http://its.acfr.usyd.edu.au/datasets/

1) Data Collection: The vehicle used to collect the data was outfitted with an ibeo.HAD Feature Fusion system [31] that provides real time detection, classification, and tracking of road users based off of lidar measurements. The lidars used are 4 beam ibeo LUX units with a field of view of 110 degrees, range of 200 meters and a recording frequency of 25 Hz. Six of these units are fitted around the bumpers of the vehicle for a complete view of the vehicle's surroundings. This vehicle can be seen in Figure 6.

The dataset used in this work was collected from 5 different roundabouts from various locations around Sydney suburbs, and are shown in Figure 5. Each of the roundabouts were chosen as typical roundabouts in Sydney suburbs, with relatively high traffic throughput. These particular roundabouts were selected as they have an approximate balance of traffic between all roads in the intersection. It is common for a roundabout to have a very infrequently used road, or are merely T junctions and not a four way intersection. The particular construction of the chosen roundabouts also allow for an unobstructed view from a vehicle near the intersection. Each recording was taken over approximately 14 hours, to capture both the morning and afternoon peaks, as this accounted for the majority of traffic across the day. Cars entering the intersection from the approach the recording vehicle was parked on, and those approaching from directly opposite the intersection were used, as the recording vehicle had significant (50 meters +) visibility down both of these roads. The 'Leith-Croydon' dataset from the authors previous work was included, but note that vehicles approaching from the East were omitted because of lesser visibility down that approach from where the recording vehicle was parked. The vehicle was parked at the first available park when leaving the roundabout. This allows the recording to emulate the visibility, and possible occlusions of a vehicle approaching the roundabout from the same direction.

Fig. 6. The vehicle used for data collection. This vehicle is outfitted with an ibeo.HAD Feature Fusion System that provides real-time detection and tracking of road users, using sensor data from six, four beam lidars.

2) Data Wrangling and Import: For result classification purposes, each recorded vehicle track is labeled as to its intersection entrance and exit, and therefore its maneuver is labeled as one in the set of [left, straight, right, u-turn]. In this way only the bounding boxes of the intersection entrances and exits are labeled, and the tracks are labeled as to which bounding box they pass through. The set of u-turns were not considered in the results, as they are exceedingly rare.

TABLE I
SUMMARY OF COLLECTED DATA

Intersection	Left	Straight	Right	U-Turn	Total
Queen-Hanks	466	5110	155	13	5744
Leith-Croydon	2577	1356	1237	16	5186
Roslyn-Crieff	183	3000	69	10	3262
Orchard-Mitchell	1716	1825	217	10	3768
Oliver-Wyndora	374	5347	222	9	5952
Total	**5316**	**16638**	**1900**	**58**	**23912**

3) Data Summary: The overall class split for each of the five intersections can be found in Table I. The intersection 'Oliver-Wyndora' was used for the test dataset, as it has the largest amount of samples in the smallest class for a four-way intersection. Using a whole intersection for the test set demonstrates the generalizability of the proposed method, as the test dataset is completely unseen during training.

B. Model Training

The model was trained using a training dataset that was balanced for each destination class using an oversampling technique. This prevents the model from biasing away from less common manoeuvres. The hyper-parameters were tuned using a grid search. A Bayesian optimizer was also used for hyper-parameter selection, but did not exceed the results of the grid search. The set of parameters with the best performance on the validation set were used, and are as follows. Adam [32] optimization was used with 0.0005 learning rate that would exponentially decay over 12 hours to a final value of 0.00001. The network used a triple stacked hyper LSTM [33] of width 256, batch size of 100, and a rate of 1.0 for α and 10.0 for β in the loss function. The number of mixtures in the MDN was set to 6. The entire sequence was sub-sampled by two, such that the sample rate becomes 12.5 Hz to reduce complexity. The encoder length was set to 7 time-steps (0.56 seconds) and the decoder was set to predict for 60 time-steps (4.8 seconds). The parameters for Multi-PAC are a threshold of 0.5, and DBSCAN was set to minimum cluster size of 1, eps distance 2 meters. The four intersections used for training data were split into a 4:1 training / validation split, and the model checkpoint used was the one with the best loss on the validation set. The Tensorflow [34] library was used for implementation.

C. Metrics

As per Section III-C, the test set data consists of only one track snippet per vehicle recorded, which is sampled at the moment the vehicle crosses the intersection entrance. The results of the proposed algorithm is compared with several other models. As the algorithm provides multi-modal output, the track with the highest probability is used, as well as the track with the most accuracy, to evaluate whether or not the model has missed the ground truth entirely, resulting in a false negative.

Commonly used is a measure in euclidean distance, which we have included. However, this metric will penalize misalignments in time and space equally, that is to say it will penalize a prediction that may have produced an incorrect speed profile (but a correct destination) with the same magnitude error as a prediction with a correct speed, but incorrect destination. This is especially true for longer (2+ seconds) prediction times. This is because at a particular time, both of these predictions are the same distance away from the ground truth. As an incorrect destination is much worse than predicting the vehicle is a car-length behind its actual position, we have included the Modified Hausdorff Distance score the account for this issue.

The metrics used are as follows:

- Total euclidean error sum - The sum of the euclidean errors between the proposed path and the ground truth, calculated for every time step.
- Horizon Euclidean error - The euclidean error between two points taken at the same time horizon, specified in seconds. The two metric Horizons lengths of 1.2 seconds and 2.8 seconds were chosen as these allow clear distinction between the left and straight sets, and the straight and right sets.
- Modified Hausdorff Distance [35] - This metric compares two tracks by considering the closest point on one line to every point on the other line. In this way it penalizes spatial divergence between two lines without penalizing temporal misalignment.

Most commonly reported is the Mean Average Error of the tracks. This is not a very good indicator, as it tends to ignore outliers and does not present the worst case scenario, especially with large amounts of data. Instead, we also include the worst 5 % and the worst 1% of scores reported for each track class to better highlight how these models may fail. Focusing on the smallest class in the test set, the right turn of Oliver-Wyndora with 222 samples, we have 11 data samples for the worst 5% and 2 samples for the worst 1%, which indicates that our dataset is large enough for these statistics to have meaning.

D. Baseline Comparisons

The following section describes the models used for comparison in the results.

- CV - The vehicle's velocity is assumed constant for the entirety of the prediction stage. The most recent 5 time-steps (0.4 seconds) are averaged and used as the constant velocity.
- CTRV - The vehicles yaw rate and velocity are assumed constant.
- CTRA - The vehicles yaw rate and acceleration are assumed constant.
- GP - A Gaussian process regression model was used to predict $\mathbf{Y}$ given $\mathbf{X}$. Due to memory constraints, only 4000 track snippets could be used for training the model. These were chosen randomly, as a properly implemented importance sampling method was not readily available. The particular implementation of GP regression was the GPy library [36].

VI. RESULTS

The results are presented in Tables II and III. These are grouped by the vehicle's destination to better illustrate how

TABLE II
CUMULATIVE RESULTS AND RESULTS GROUPED BY VEHICLES TURNING RIGHT

	All tracks								Right turns only							
Metric	CTRA	CTRV	CV	GP	RNN-FF Best	RNN-FF Selected	RNN-ZF Selected	RNN-FL Selected	CTRA	CTRV	CV	GP	RNN-FF Best	RNN-FF Selected	RNN-ZF Selected	RNN-FL Selected
MHD mean	2.66	2.65	2.04	1.11	0.71	**0.83**	0.93	6.63	7.34	8.00	7.18	4.56	1.15	**1.48**	1.51	6.63
MHD worst 5%	7.28	8.25	4.01	3.77	1.76	**2.09**	2.16	8.04	11.82	11.58	10.67	6.62	2.76	**3.78**	3.84	8.26
MHD worst 1%	10.36	10.44	9.07	5.79	2.74	**5.74**	5.95	8.56	18.27	13.55	11.36	7.83	3.34	**6.11**	6.13	8.57
Euclidean mean	2.94	2.95	2.30	1.45	1.18	**1.34**	1.45	7.32	9.05	9.84	8.46	5.15	1.90	**2.21**	2.28	7.17
Euclidean worst 5%	9.13	9.45	5.01	4.43	2.80	**3.37**	3.80	8.76	14.87	15.84	12.62	7.29	3.80	**4.96**	5.17	8.90
Euclidean worst 1%	11.58	11.99	10.44	6.57	4.31	**6.48**	6.58	9.26	20.51	17.92	13.36	8.75	5.13	6.90	**6.79**	9.13
Horizon 2.8s mean	8.03	7.98	6.34	4.70	3.14	**4.00**	4.13	13.44	9.76	10.59	8.80	5.76	2.12	3.28	**2.96**	8.85
Horizon 2.8s worst 5%	17.18	15.79	11.23	**9.31**	7.77	9.66	9.54	17.28	16.54	18.87	13.86	8.77	5.38	8.49	**8.10**	10.91
Horizon 2.8s worst 1%	21.64	20.59	14.25	14.00	9.80	**13.54**	15.00	18.05	21.83	21.91	16.07	11.63	6.89	9.26	**8.84**	11.22

TABLE III
RESULTS GROUPED BY STRAIGHT AND LEFT TURNING VEHICLES

	Straight tracks								Left turns only							
Metric	CTRA	CTRV	CV	GP	RNN-FF Best	RNN-FF Selected	RNN-ZF Selected	RNN-FL Selected	CTRA	CTRV	CV	GP	RNN-FF Best	RNN-FF Selected	RNN-ZF Selected	RNN-FL Selected
MHD mean	2.53	2.49	1.79	0.94	0.68	**0.78**	0.90	6.85	1.71	1.67	2.39	1.41	1.01	1.18	**1.04**	3.60
MHD worst 5%	5.93	5.50	2.88	2.60	1.48	**1.90**	1.98	8.04	3.65	3.73	3.61	2.76	2.47	3.05	**2.67**	4.23
MHD worst 1%	10.24	10.21	**4.51**	5.07	2.55	5.81	6.07	8.57	5.99	5.28	4.79	4.10	3.27	**3.65**	3.73	4.59
Euclidean mean	2.75	2.73	2.02	**1.29**	1.14	**1.29**	1.42	7.59	1.85	1.79	2.49	1.54	1.41	1.48	**1.32**	3.76
Euclidean worst 5%	6.99	6.10	3.76	3.35	2.67	**3.15**	3.70	8.78	4.54	3.94	3.86	**3.08**	3.40	3.53	3.27	4.43
Euclidean worst 1%	11.08	10.88	**5.63**	5.64	4.28	6.48	6.60	9.27	6.12	5.61	4.83	4.28	3.89	**4.14**	4.32	4.96
Horizon 1.2s mean	2.06	2.08	1.50	**1.03**	1.05	1.17	1.32	7.13	2.25	2.18	3.05	1.85	1.63	1.70	**1.36**	4.69
Horizon 1.2s worst 5%	4.94	4.81	2.82	**2.45**	2.25	2.79	3.03	10.55	4.00	3.72	4.49	**4.06**	5.10	5.28	4.29	7.04
Horizon 1.2s worst 1%	11.81	11.56	4.28	**4.33**	3.66	4.93	5.82	11.88	9.02	7.69	5.52	**5.05**	6.12	6.16	5.36	7.81

each method handles vehicles traveling in particular directions, as well as handling the large class imbalance when considering all tracks at once. A sample of each class is presented in Figure 7, as well as a failure case.

A. Quantitative Results

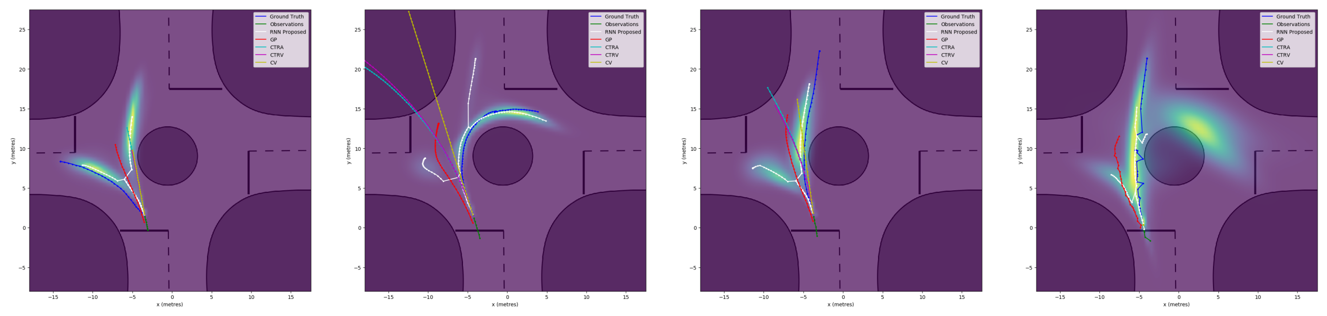
The results of the models and the baseline comparisons are presented in Tables II and III. For the RNN-FF model, the most probable path is presented, as well as the path with the lowest error. This highlights whether or not the model had completely missed the proper prediction, resulting in a false negative. Note that the horizon time for Table III is 1.2 seconds, as this is the time for a track where the divergence between vehicles traveling straight and turning right is apparent. Similarly, a horizon time of 2.8 seconds was chosen for Table II for the separation of vehicles turning right and traveling straight ahead.

The most probable path the RNN-FF model proposed outperforms all baselines, especially for vehicles turning either left or right. One intersecting outlier for this is the MHD results for the worst 5% and 1% of vehicles turning left. Remember, the MHD metric penalizes tracks that are dissimilar in space, but not in time. In these cases, the RNN-FF algorithm has proposed that the vehicle is making a different turn, and the prediction differs greatly from the ground truth. Some of the GP output was the average of two tracks, which was not a valid direction of travel for this intersection. But given that its proposal is closer to the ground truth, it scores higher for this metric, even though the proposed vehicle trajectory was invalid. Even considering these cases, the best scoring path the RNN-FF model proposed outperforms any baseline, demonstrating that the ground truth was contained in the multi-modal set of predictions, albeit with a lower probability.

B. Qualitative Results

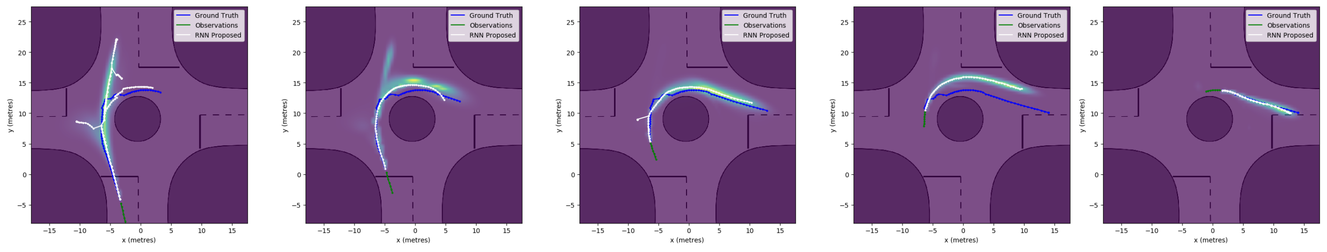
Figure 7 shows the performance of the chosen RNN-FF model and the baseline models. The multiple paths the RNN model has predicted are depicted in white, the observations in green, and the ground truth of the vehicle in blue. The ground truth spans for about six seconds. Here is where the multi-modal nature of the output distribution can be clearly observed. For the vehicle traveling straight, in Figure 7c, there is still some probability that the vehicle will travel left, and so the model has produced a second hypothesis to convey this. Note that the most confident prediction is still straight ahead, which is the correct prediction. This model behavior can also be seen in Figure 7a, where a vehicle is turning left. The final case presented is where a vehicle had come to a complete stop, which makes for a very difficult prediction. Here the algorithm produces significant uncertainty for all directions. The amount of noise that exists in the data can also be seen quite clearly here, with significant jumps in the ground truth data.

A single vehicle traversing the intersection can be observed in Figure 8. When a vehicle is far from the intersection,



(a) (b) (c) (d)

Fig. 7. Results from four different vehicles traveling in different directions: left 7a, right 7b, straight 7c, 7d. Here the output of the RNN-FF network is shown as a heatmap, and the multi-modal clustered output is shown in white. The baseline models are also depicted. In the final figure 7d, the vehicle has stopped before entering the intersection, which makes intention prediction very difficult.



(a) (b) (c) (d) (e)

Fig. 8. Results of a single vehicle turning right at several different distances traveled past the reference line. The distances are: 8a: −5 meters, 8b: 0 meters, 8c: 5 meters, 8d: 10 meters 8e: 20 meters. This sequence of a single vehicle traversing the intersection demonstrates how the distribution converges to the final solution when enough time has passed to make the vehicle's intention obvious. Note the correction for data misalignment between (d) and (e).

the driver has not yet committed to taking any particular maneuver at the intersection. In this case, all maneuvers are possible, and the prediction algorithm suggests three possible paths, seen in Figure 8a.

Once the driver enters the intersection, it is traveling too quickly for a left turn to be feasible, and the algorithm weights the right turn more heavily, seen in Figure 8b. The model shows stronger confidence in a right turn after the vehicle has traveled 10 meters past the intersection entrance, depicted in Figure 8c. As this test data is on a completely different intersection than those in the training set, misalignments in the data exist, and this becomes visible in Figure 8d. Here, the algorithm is still predicting a right turn, but the exit is not exactly where it was in previous sequences. This bias is corrected for with slightly more data, as seen in Figure 8e.

C. Ablative Results

The RNN-FL model produced the worst results, matching scores to that of the CTRA model. This demonstrates the need to do complete forward path prediction at training time, instead of using loss on the first time-step. Interestingly, the differences between the RNN-ZF model and the RNN-FF model were minimal. This suggests that the single sample taken at training or inference time that is used as input for the next time-step for the prediction is not very important. This can be attributed to how taking a single sample for each time-step from a multi-modal distribution produced a very noisy output, as it effectively considered the vehicle to be in two modalities at once. This means that the RNN is not only sensitive to the input data, but also how many recurrences of the RNN have already occurred, as it begins to ignore the input data after a set number of recurrences.

VII. Conclusion

In this paper we have presented a method for driver intention and path prediction with a multi-modal, probabilistic solution. This is achieved through the use of recurrent neural networks combined with a mixture density network output function. This output is passed through a clustering algorithm to produce a ranked set of possible trajectories, each with uncertainty. A naturalistic, real-world dataset was taken to validate the results, resulting in the collection of over 23,000 vehicles traveling through five different roundabouts. To the author's knowledge, this is the largest publicly available dataset of its kind. The algorithm was tested on 5952 real-world trajectories, and outperformed all baselines.

All code used for data wrangling, model training / testing, graph and table generation can be found in the code repository - Recurrent Architecture for Driver Intention Prediction (RADIP) at: https://github.com/azyner/radip The dataset is available at: http://its.acfr.usyd.edu.au/datasets/

References

[1] V. Alexiadis, J. Colyar, J. Halkias, R. Hranac, and G. McHale, "The next generation simulation program," *ITE J. Inst. Transp. Eng.*, vol. 74, no. 8, p. 22, 2004.

[2] E.-H. Choi, *Crash Factors in Intersection-Related Crashes: An On-Scene Perspective*, Nat. Highway Traffic Saf. Admin., U.S. Dept. Transp., Washington, DC, USA, Tech. Rep. DOT HS 811 366, 2010. [Online]. Available: https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/811366

[3] P. Kumar, M. Perrollaz, S. Lefèvre, and C. Laugier, "Learning-based approach for online lane change intention prediction," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2013, pp. 797–802.

[4] T. Streubel and K. H. Hoffmann, "Prediction of driver intended path at intersections," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2014, pp. 134–139.

[5] M. Schreier, V. Willert, and J. Adamy, "Bayesian, maneuver-based, long-term trajectory prediction and criticality assessment for driver assistance systems," in *Proc. Int. IEEE Conf. Intell. Transp. Syst.*, Oct. 2014, pp. 334–341.

[6] A. Zyner, S. Worrall, and E. Nebot, "A recurrent neural network solution for predicting driver intention at unsignalized intersections," *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 1759–1764, Jul. 2018.

[7] D. J. Phillips, T. A. Wheeler, and M. J. Kochenderfer, "Generalizable intention prediction of human drivers at intersections," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2017, pp. 1665–1670.

[8] H. Veeraraghavan, N. Papanikolopoulos, and P. Schrater, "Deterministic sampling-based switching Kalman filtering for vehicle tracking," in *Proc. IEEE Intell. Transp. Syst. Conf.*, Sep. 2006, pp. 1340–1345.

[9] M. Althoff and A. Mergel, "Comparison of Markov chain abstraction and monte carlo simulation for the safety assessment of autonomous cars," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 1237–1247, Dec. 2011.

[10] J. Wiest, M. Höffken, U. Kreβel, and K. Dietmayer, "Probabilistic trajectory prediction with Gaussian mixture models," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2012, pp. 141–146.

[11] Q. Tran and J. Firl, "Online maneuver recognition and multimodal trajectory prediction for intersection assistance using nonparametric regression," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2014, pp. 918–923.

[12] N. Deo and M. M. Trivedi. (May 2018). "Multi-modal trajectory prediction of surrounding vehicles with maneuver based lstms." [Online]. Available: https://arxiv.org/abs/1805.05499

[13] S. H. Park, B. Kim, C. M. Kang, C. C. Chung, and J. W. Choi. (Feb. 2018). "Sequence-to-sequence prediction of vehicle trajectory via LSTM encoder-decoder architecture." [Online]. Available: https://arxiv.org/abs/1802.06338

[14] A. Kuefler, J. Morton, T. Wheeler, and M. Kochenderfer, "Imitating driver behavior with generative adversarial networks," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2017, pp. 204–211.

[15] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. S. Torr, and M. Chandraker, "DESIRE: Distant future prediction in dynamic scenes with interacting agents," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 336–345.

[16] E. Schmerling, K. Leung, W. Vollprecht, and M. Pavone. (Oct. 2017). "Multimodal probabilistic model-based planning for human-robot interaction." [Online]. Available: https://arxiv.org/abs/1710.09483

[17] A. Robicquet, A. Sadeghian, A. Alahi, and S. Savarese, "Learning social etiquette: Human trajectory understanding in crowded scenes," in *Proc. Eur. Conf. Comput. Vis.* New York, NY, USA: Springer, Oct. 2016, pp. 549–565.

[18] M. Schreier, V. Willert, and J. Adamy, "An integrated approach to maneuver-based trajectory prediction and criticality assessment in arbitrary road environments," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 10, pp. 2751–2766, Oct. 2016.

[19] H. Rouzikhah, M. King, and A. Rakotonirainy, "The validity of simulators in studying driving behaviours," in *Proc. Australas. Road Saf. Res., Policing Educ. Conf.*, Sep. 2010, p. 1.

[20] C. Thiemann, M. Treiber, and A. Kesting, "Estimating acceleration and lane-changing dynamics from next generation simulation trajectory data," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2088, no. 1, pp. 90–101, Jan. 2008.

[21] B. Coifman and L. Li, "A critical evaluation of the next generation simulation (NGSIM) vehicle trajectory dataset," *Transp. Res. B, Methodol.*, vol. 105, pp. 362–377, Nov. 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0191261517300838

[22] A. Zyner, S. Worrall, J. Ward, and E. Nebot, "Long short term memory for driver intent prediction," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2017, pp. 1484–1489.

[23] A. Graves. (Aug. 2013). "Generating sequences with recurrent neural networks." [Online]. Available: https://arxiv.org/abs/1308.0850

[24] D. Ha and D. Eck. (Apr. 2017). "A neural representation of sketch drawings." [Online]. Available: https://arxiv.org/abs/1704.03477

[25] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human trajectory prediction in crowded spaces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 961–971.

[26] M. Pfeiffer, G. Paolo, H. Sommer, J. Nieto, R. Siegwart, and C. Cadena. (Sep. 2017). "A data-driven model for interaction-aware pedestrian motion prediction in object cluttered environments." [Online]. Available: https://arxiv.org/abs/1709.08528

[27] N. Deo and M. M. Trivedi. (May 2018). "Convolutional social pooling for vehicle trajectory prediction." [Online]. Available: https://arxiv.org/abs/1805.06771

[28] E. Strigel, D. Meissner, F. Seeliger, B. Wilking, and K. Dietmayer, "The Ko-PER intersection laserscanner and video dataset," in *Proc. 17th Int. IEEE Conf. Intell. Transp. Syst.*, Oct. 2014, pp. 1900–1901.

[29] A. Khosroshahi, E. Ohn-Bar, and M. M. Trivedi, "Surround vehicles trajectory analysis with recurrent neural networks," in *Proc. IEEE 19th Int. Conf. Intell. Transp. Syst.*, Nov. 2016, pp. 2267–2272.

[30] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd Int. Conf. Knowl. Discovery Data Mining*, Aug. 1996, pp. 226–231.

[31] *Ibeo Automotive Systems GmbH*. Accessed: May 2, 2018. [Online]. Available: www.ibeo-as.de

[32] D. P. Kingma and J. Ba. (Dec. 2014). "Adam: A method for stochastic optimization." [Online]. Available: https://arxiv.org/abs/1412.6980

[33] D. Ha, A. Dai, and Q. V. Le. (Sep. 2016). "HyperNetworks." [Online]. Available: https://arxiv.org/abs/1609.09106

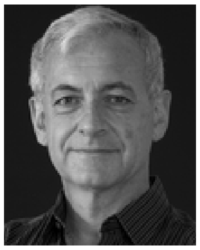
[34] M. Abadi. (Mar. 2016). "TensorFlow: Large-scale machine learning on heterogeneous distributed systems." [Online]. Available: https://arxiv.org/abs/1603.04467

[35] M.-P. Dubuisson and A. K. Jain, "A modified Hausdorff distance for object matching," in *Proc. 12th IAPR Int. Conf. Pattern Recognit. (ICPR)*, vol. 1. Oct. 1994, pp. 566–568.

[36] GPy. (2012). *GPy: A Gaussian Process Framework in Python*. [Online]. Available: http://github.com/SheffieldML/GPy

Alex Zyner received the B.E. (Hons.) degree in mechatronics/B.Sc. degree from The University of Sydney, Australia, in 2014, where he is currently pursuing the Ph.D. degree. His research is focused on motion prediction for vehicles to improve autonomous car safety.

Stewart Worrall received the Ph.D. degree from The University of Sydney, Australia, in 2009, where he is currently a Research Fellow with the Australian Centre for Field Robotics. His research is focused on the study and application of technology for vehicle automation and improving safety.



Eduardo Nebot received the B.Sc. degree in electrical engineering from the Universidad Nacional del Sur, Argentina, the M.Sc. and Ph.D. degrees from Colorado State University, Colorado, USA. He is currently a Professor with The University of Sydney, Sydney, Australia, and the Director of the Australian Centre for Field Robotics. His main research interests include robotics automation and intelligent transport systems. The major impact of his fundamental research is in autonomous systems, navigation, and safety.