2018 IEEE International Conference on Robotics and Automation (ICRA)
May 21-25, 2018, Brisbane, Australia

Predicting Ego-Vehicle Paths from Environmental Observations with a Deep Neural Network

Ulrich Baumann[1], Claudius Gläser[1], Michael Herman[1] and J. Marius Zöllner[2]

***Abstract*— Advanced driver assistance systems allow for increasing user comfort and safety by sensing the environment and anticipating upcoming hazards. Often, this requires to accurately predict how situations will change. Recent approaches make simplifying assumptions on the predictive model of the Ego-Vehicle motion or assume prior knowledge, such as road topologies, to be available. However, in many urban areas this assumption is not satisfied. Furthermore, temporary changes (e.g. construction areas, vehicles parked on the street) are not considered by such models. Since many cars observe the environment with several different sensors, predictive models can benefit from them by considering environmental properties. In this work, we present an approach for an Ego-Vehicle path prediction from such sensor measurements of the static vehicle environment. Besides proposing a learned model for predicting the driver's multi-modal future path as a grid-based prediction, we derive an approach for extracting paths from it. In driver assistance systems both can be used to solve varying assistance tasks. The proposed approach is evaluated on real driving data and outperforms several baseline approaches.**

I. Introduction

An accurate prediction of the Ego-Vehicle motion is a key ingredient for many advanced driver assistant systems (ADAS) that execute evasive actions to avoid or mitigate an upcoming hazard. Such actions could be an emergency stop or an evasive steering maneuver and thus directly influence the vehicle's behavior. As a consequence, they can pose a severe risk to the Ego-Vehicle and its environment. Therefore, such ADAS require high confidence detections of upcoming hazards. Besides a high accuracy, these situations need to be detected early enough such that successful evasive strategies still exist. In order to meet both requirements, it is crucial to predict the motion of the Ego-Vehicle precisely.

In the context of driver assistant systems, the human driver is in full charge of the Ego-Vehicle motion: He selects the destination, the route leading to his destination and controls the vehicle. While the vehicle control signals can be measured, the driver's intent (route and destination) is often not known and cannot be directly observed. As a consequence, a predictive model of the future Ego-Vehicle motion needs to infer the driver's intent from the history of observed sensor measurements. For this task, two different sources of data are of special interest: data describing the physical state of the vehicle itself (e.g. pose, velocity, acceleration, etc.) as well as the environment around the vehicle (e.g. roads, free space, curb stones, etc.). Human drivers usually refrain from changing the physical state of the vehicle rapidly, which allows to use the extrapolation of the current physical state in time as a prediction for the future vehicle motion. Various approaches, which are summarized in [1], have been proposed to predict the vehicle motion based on this principle. Such models typically assume a constant turn rate and a constant velocity or acceleration. Since these approaches are, due to their simplicity, very common we benchmark the proposed approach from this work against these baseline approaches. While such physically motivated models might be appropriate for short prediction horizons, they are typically not meaningful for long horizons, since assumptions are not met anymore. For example, they have limitations in predicting changes caused by driving maneuvers, such as turns. Since a human driver's behavior is strongly influenced by the environment, such as road topology, lanes, sight, traffic regulations or other traffic participants, utilizing this type of environmental information can improve the prediction accuracy. An example for a situation where environmental data is beneficial is given in Fig. 1. Here, the Ego-Vehicle approaches a fork with a constant velocity and vanishing turn rate. Hence, there are no hints for the upcoming turn in the physical state of the vehicle, while it is obvious in the environmental data. Other approaches tackle this by assuming that global knowledge

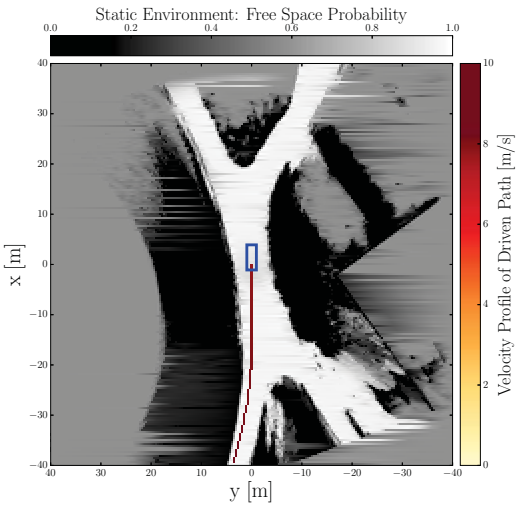


Fig. 1: An example of an urban driving scenario with additional information on the future Ego-Vehicle motion provided by its static environment. The Ego-Vehicle (blue box) is positioned in the center of the grid heading to the top. Its static environment is represented in shades of gray depending on the measured value of each grid cell's probability for being free space, while the red line indicates the previously driven path color-coded with respect to its velocity profile.

[1] Robert Bosch GmbH, Corporate Research, 71272 Renningen, Germany.
[2] Research Center for Information Technology (FZI), 76131 Karlsruhe, Germany.

is available such as the road topology including lanes. However, in urban settings with local, temporary obstacles these models might make inaccurate predictions. While the environment does provide additional valuable information for the prediction, this information might not be able to fully resolve ambiguities. This is mainly caused by unavailable information, such as the unknown intent of the driver. Some of the missing information might be inferred to some extent, but not all of it might be available. Unresolved discrete ambiguities are referred to as modes within this paper. In the example of Fig. 1 environmental data introduced two modes: following the left, or right road. For high quality predictions, it is necessary to have a model that can represent and therefore handle such multi-modal situations.

In this work, we propose a learned model that makes accurate predictions of future positions of an Ego-Vehicle, which is controlled by a human driver, solely from sensor measurements (LiDAR) of the static environment and the past path. The model can be easily extended to other modalities, if the sensor measurement can be transformed into an appropriate grid-based representation. The grid-based prediction of the future path, which is from here on denoted as the *prediction map*, can be directly used for driver assistance tasks. Since some problems require the prediction of actual paths instead of the *prediction maps*, we further propose a method for extracting paths from them. The system is evaluated on real world data that has been recorded in an urban setting. We chose an urban setting, since it is unstructured and versatile and thus promising to explore the advantages of environmentally aware path predictions.

Key contribution of this work is a model that predicts the probability of a driver to visit a particular state, based on his previous path as well as sensor measurements. This state-dependent probability can be represented as a grid, the *prediction map*. The model is fully learned and is able to predict nonlinear, complex behavior that incorporates information of the static environment around the Ego-Vehicle. The proposed *prediction map* method is further able to provide multi-modal predictions, which can be used to extract distinct options that human drivers have while driving (e.g. discrete choices at intersections). Since the model is based on data that can be acquired from on-board sensors, it can be directly applied to real world problems without requiring access to external or global data sources. An evaluation on a real world dataset has shown that previous approaches are outperformed by a significant margin. The largest improvement is obtained when the environment has a large influence on driving behavior.

II. Related Work

There is a wide range of models for the prediction of a vehicle's future motion, varying in both, the used methods and the kind of data they use as an input. One class of models predict future motion solely from measurements of the vehicle's physical state. For example, in [2] and [3] kinematic motion models are used to extrapolate trajectories. These kinematic models make assumptions on the future motion of the vehicles, which are, for the two given examples, a *constant acceleration* and a *constant turn rate and acceleration* (*CTRA*), respectively. However, these models tend to perform poorly when these assumptions are violated, which is often the case during driving maneuvers. Other models extend this approach and consider a sequence of recorded physical states. They consequently base their predictions on the recently driven trajectory, which allows to identify driving maneuvers and adjust the predictions. In [4], demonstrations are used to learn a joint probability distribution over trajectories, which can be used as a motion model for predictions. In contrast, the approach in [5] proposes to cluster demonstrated trajectories and then to predict trajectories according to the cluster mean that is closest to the observation. A conceptual related approach is taken in [6], where long term trajectory predictions are made with a trajectory classification and a particle filter framework. All these models have in common, that they solely operate on data describing the current physical state of the vehicle or its history. Thus, they cannot adjust the prediction to a driving maneuver like turning, until the maneuver can be detected from the vehicle's physical state.

To overcome this limitation it is beneficial to consider additional information in the predictions. This is done for example in [7] that utilizes a Bayesian network to predict the maneuver intent at intersections, based on map data and turn signals. Similar to all map-based approaches it is limited by the availability and quality of high precision up-to-date maps. One possibility to avoid this issue is to use additional data that is observed by sensors. For example, [8] utilizes lane information to improve predictions from a *CTRA* kinematic motion model. This approach is however limited to scenarios with well defined lanes, and therefore less suited for urban settings, where the existence of predefined lanes is not guaranteed. In [9], vehicle paths are predicted by combining a kinematic motion model with data on road boundaries measured with radar. Two independent paths are predicted, one based on the motion model and one based on the road boundary measurements, and then fused to the final path prediction. While this approach considers measured data on the static vehicle environment, it does a priori not allow for multi-modal predictions in ambiguous situation like the approach proposed in this paper.

In recent publications, artificial neural networks were used for motion prediction. Two such approaches, that are closely related to the work presented in this paper, are [10] and [11]. Both use convolutional neural networks to extract path proposals from environmental observations recorded with a monocular camera or a LiDAR scanner, respectively. In [10] path proposals are inferred directly in image space by semantically labeling pixels, but the path proposals are not transferred to drivable path predictions in the world or vehicle frame. A grid-based representation of the path predictions in the vehicle frame, similar to the one in this paper, is used in [11]. This approach utilizes additional information e.g. on the drivers intent (go left, straight or right) to resolve ambiguities in the path predictions at intersections.

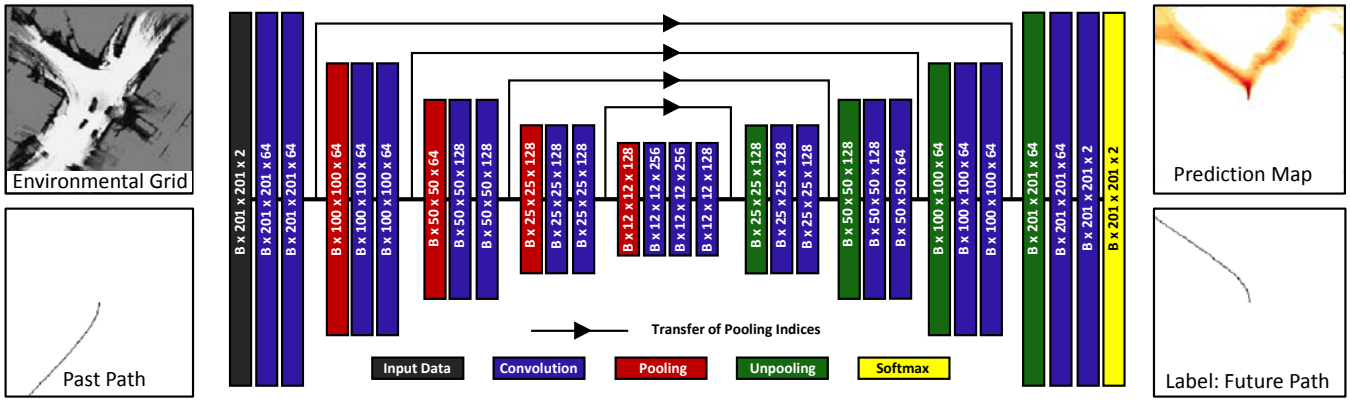


Fig. 2: Schematic illustration of the architecture of the proposed CNN with examples of the input channels, the *prediction map* representing the path prediction and the ground truth label. The output of each functional layer of the network is represented by a rectangle color-coded according to the type of layer that produced it. The shape of the output data structures is written on each rectangle in the following format: Batch Size (B) x Height x Width x Channel. Black arrows represent the transfer of the stored pooling indices from each pooling layer to their corresponding unpoolig layer.

However, this information is typically not available within the context of ADAS as long as the intent is not estimated with other methods. Furthermore, [12] proposes a recurrent neural network that predicts vehicle trajectories in highway scenarios. Here, the prediction of the future motion is also represented in a grid, but it is solely based on kinematic data of all sensed vehicles and therefore does not use information on the static environment.

III. Learning a Model for Path Prediction

A. Representation of Data

The proposed path prediction model utilizes the vehicle's observation of the static environment around the car as well as the previously driven path. Data on both is collected within a monitored square area covering 40 meter to the front, back, left and right of the Ego-Vehicle and is stored in two two-dimensional grids. These grids are composed of 201 times 201 equally sized grid cells and represent the x-y plane of the monitored area. Coordinates are defined in the vehicle body frame by a right-handed orthogonal coordinate system, where the x-axis points towards the front of the vehicle and the y-axis points to the left. The past path is represented in a binary grid, where each grid cell is either assigned the value 1, if the previously driven path passes through the area corresponding to this grid cell or 0 if it does not. Information on the static environment of the vehicle is stored in the second grid, where each grid cell is assigned a continuous value between 0 and 1 according to its measured probability for being free space, while unobserved cells are set to 0.5. The predicted path is represented by the *prediction map*. It is a grid of the same size as the past path grid and continuous values between 0 and 1 are assigned to the grid cells according to the model's prediction. A value of 1 (0) corresponds to the maximal (minimal) prediction confidence for the cell being part of the path. Ground truth labels for the supervised learning problems can be easily generated from recorded data, since the driven path in the future can be used as a target. The future path is represented identically to the past path. An example for the two input grids, the label and the *prediction map* is shown, together with the model architecture, in Fig. 2.

B. Model

Given the selected representation of the input data and the predicted path, the task for the model is to map the two two-dimensional input grids containing spatially correlated data to cell-wise predictions of the path stored in a grid of identical size. This corresponds to a cell-wise classification problem with two classes: does the grid cell belong to the path or does it not, that are denoted by *path* and *not-path*, respectively. The structure of the problem is therefore closely related to the well studied machine learning problem of semantic segmentation of images, where an assignment of a semantic label is learned for each pixel of the image. Typical architectures of neural nets used for the semantic segmentation are convolutional encoder-decoder structures. They first learn an abstract, compact, and low-dimensional representation of the input data via an encoder network, which reduces the spatial resolution. Then, a decoder is utilized to learn a mapping from this representation to the final output, which has the original spatial resolution.

The proposed architecture is a convolutional feed forward neural network (CNN) that follows this paradigm as it is designed as a symmetric encoder-decoder structure. It is build with four types of functional layers: convolutional layers, 2x2 max pooling layers, 2x2 unpooling layers and a softmax output layer. All convolutional layers are two-dimensional convolutions with a kernel of 3x3 and a stride of 1. They are all, except the last one, followed by a batch normalization [13] and ReLU activation [14]. The output of the proposed architecture are two grids, one for each class, of the same spatial resolution as the input that are normalized using a softmax function over the two classes within each grid cell. Furthermore, symmetric padding is applied to the spatial dimensions of all convolutional layers and to the first and last unpooling layer to match spatial resolutions. Fig. 2 gives an overview of the proposed architecture and lists all functional layers with their output data structure resolution.

The proposed model uses the same architectural concept as SegNet [15], but some adjustment for the given problem were made: The total number of free parameters (approximately 2.5 million) is kept smaller to control overfitting and the depth of the network was set such that the receptive field of each grid cell in the output covers the whole input grid.

C. Training

The proposed network is trained from scratch starting by initializing all convolutional filters according to the so-called *Xavier* initialization routine [16], while initializing all learned means and variances of the batch normalization to 0 and 1, respectively. It is trained in a supervised fashion by minimizing the loss of the training dataset by stochastic gradient descent with Adam [17] as an optimizer. The loss function L is composed of a weighted sum of the cross-entropy loss L_{CE} of each grid cell and a L2 regularization loss. It is given by

$$L = \sum_{c \in C} f(c) L_{\mathrm{CE}}(c) + \lambda \sum_{w \in W} w^2, \tag{1}$$

using the set of all grid cells C and all weights W. The cross-entropy loss weighting $f(c)$ is defined as

$$f(c) = \begin{cases} \gamma & \text{ground truth class of cell is } \textit{path} \\ 1 & \text{otherwise} \end{cases} \tag{2}$$

and is used to scale the contribution of grid cells, that have the ground truth class *path*, by a factor of γ to the cross-entropy loss. This is necessary, as in a typical scenario, there are roughly 400 times more grid cells of the ground truth class *not-path* than *path*, yielding a strong bias in the prediction towards the class *not-path*. Additionally, the scale factor λ was introduced to balance the contributions of the cross-entropy loss and the regularization loss to adjust the strength of the regularization. Learning rate decay is applied to reduce the learning rate of the Adam optimizer continuously, as the training progresses. Optimal values for the so-called hyperparameters, being the decay factor of the learning rate *lrd*, the learning rate *lr* itself and the scale factors γ and λ, have to be determined experimentally by a hyperparameter optimization.

IV. Extracting Paths from Prediction Maps

While the *prediction maps* indicate which grid cells are likely a part of the path, they do not predict an actual path in the sense of a one-dimensional line in the x-y-plane. Furthermore, there is no guarantee that the set of all grid cells, that are marked in the *prediction maps* as part of the path, connects the center (current Ego-Vehicle position) with the edge of the grid in a coherent and drivable fashion. To overcome this shortcoming, a method is introduced to extract one-dimensional, coherent and drivable paths from the *prediction maps* in four steps. An example for a path extracted by the proposed method, is given in Fig. 3. In the first step of the path extraction, the *prediction map* is transformed to a local cost map, given by 1 minus the *prediction map*. The local cost map is then used to calculate a

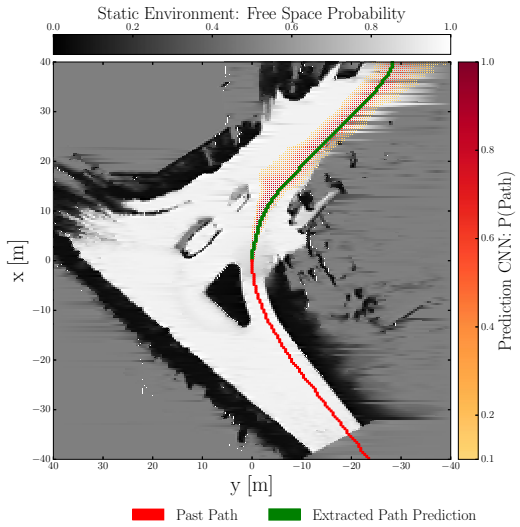


Fig. 3: Example of a predicted path, extracted by the proposed method. The green line represents the path extracted from the *prediction map*, that is shown in shades of orange and red, while the red line corresponds to the past path.

global cost map, that contains the minimal cost for reaching a cell from the grid center, by considering the local cost of all cells. This second step is necessary, as the decision on whether a cell is part of the path or not should be based on the full prediction of the model and not just on the local prediction for a cell. The global cost of a cell is given by the sum of the smallest global cost of a neighboring cell, the local cost of the cell itself and a small transition cost that is introduced to stabilize the algorithm. The algorithm that is used to calculate a global from a local cost map is given in Algorithm 1. In a third step, a greedy path is extracted

Algorithm 1 Global cost map computation

```
transition_cost ← 10^−6
first_cell ← Cell of grid center
global_cost_map: Initialize to 10^9 for all cells
global_cost_map[first_cell] ← local_cost_map[first_cell]
q: Queue for storing grid cells
q.add(first_cell)
while q.is_not_empty() do
    c ← q.get_next_cell()
    neighbor_list ← neighboring cells of c
    for all c_n ∈ neighbor_list do
        cost ← global_cost_map[c] + local_cost_map[c_n]
        cost ← cost + transition_cost
        if cost < global_cost_map[c_n] then
            global_cost_map[c_n] ← cost
            q.add(c_n)
return global_cost_map
```

from the global cost map, which is seeded with the cell on the edge of the grid that has the lowest global cost. Starting with that seed, the neighboring cell with the lowest global cost is selected as part of the greedy path. This is repeated sequentially with the newly selected cell until the center of

the grid is reached, which has by definition of Algorithm 1 the smallest global cost of all grid cells. Since there is no guarantee that the greedy path is drivable, as it might contain turns with a larger curvature than the Ego-Vehicle is capable to drive, greedy paths are smoothened in a forth step. During this process, a smooth path is extracted from the center to the edge of the grid, by following the greedy path with some constraints on the curvature of the path. These constraints ensure that the maximal drivable curvature of the path is not exceeded, and that the curvature does not change rapidly. It is however not meaningful to continue this procedure and follow a change of direction indicated by the greedy path, when the prediction confidence for the class *path* is very low on the remainder of the greedy path. Therefore, a straight path is assumed, once the mean prediction for the class *path* drops below a threshold for the remainder of the greedy path.

This procedure extracts a path for the mode with the lowest global cost for reaching the edge of the grid. As global costs are calculated for all grid cells within step two, it is furthermore possible to extract paths for the other modes by selecting the proper seeds for the greedy path extraction, which correspond to local minima in the global cost of the edge cells. However, if there is no interest in extracting the full global cost map or paths for additional modes, step two and three can be replaced by a shortest path search algorithm. For example, the computationally more efficient A* algorithm [18] yields equivalent results, if an appropriate heuristic is used, such as the Manhattan distance to the grid edge times the transition costs.

V. Experimental Evaluation

We evaluate the proposed approach based on a real world dataset that has been recorded with a car. A supplementary video, that is available online[1], provides a deeper insight into the model's performance.

A. Dataset

The dataset, which is split in 52 sequences, consists of 2.5 hours of real world driving data recorded in urban settings in Germany. Raw LiDAR measurements are converted to the grid representations of the static vehicle environment, while the vehicle position is inferred from odometry and interpolated to receive dense paths in the grids. Both, the vehicle's position and the environmental grid, are saved with a frame rate of 10 Hz. Frames at the start and the end of each sequence are neglected to ensure that there is no overlap between sequences. Furthermore, all frames recorded with a vehicle velocity of 0 m/s are neglected, to avoid having identical or highly correlated samples in the dataset. The full dataset is then split into a training, validation and test dataset with an anticipated ratio of 0.6, 0.2 and 0.2, respectively, by randomly assigning the sequences to the datasets. This resulted in distributing 64.6 %, 18.2 % and 18.2 % of the total 67578 used frames to the respective datasets.

[1] https://youtu.be/h21Exb3XPn8

B. Data Augmentation

Since the proposed architecture has a large number of free tunable parameters, it is prone to overfitting. A method to overcome this burden of small datasets is data augmentation, which is heavily used in many deep learning domains. Two methods of data augmentation were applied: First, the environmental grid as well as the past and future path are rotated by an angle uniformly sampled from $[0, 2\pi)$. While the environmental grid is rotated directly, the past and future path (ground truth label) are rotated in their coordinate representation and then transformed to grids to avoid smearing effects. In a second step, a cell-wise sampled noise is added to all grid cells of the environmental grid, that are unlikely to be free space $P(\text{Free Space}) \leq 0.5$, by

$$P(\text{Free Space}) \mapsto P(\text{Free Space}) + A * N_c. \tag{3}$$

Here, A denotes a grid-wide used amplitude that is sampled uniformly from $[0, 1]$ and N_c is a cell noise that is sampled for each grid cell from the uniform distribution on $[-0.125, 0.125]$. The grid-wide amplitude A was introduced to ensure that the network is trained on both, training examples close to the unaugmented data ($A \approx 0$) and strongly augmented data ($A \approx 1$). Finally, all values of $P(\text{Free Space}) < 0$ are set to 0 for consistency.

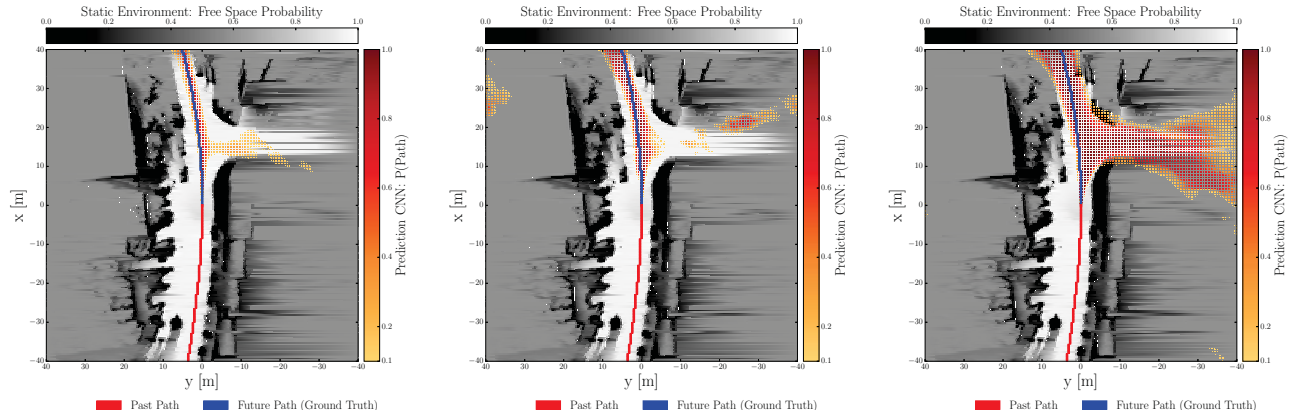
C. Evaluation of the Prediction Maps

Table I gives an overview of the hyperparameters that have been tested in a grid search manner, executed in two optimization blocks. Within each block, a model was trained from scratch for each possible combination of hyperparameters. The second optimization block was found to be necessary, as the best performing model for the path extraction was found to be at the edge of the tested range for three out of the four hyperparameters. Early stopping based on the validation set was applied to prevent overfitting to the training set.

TABLE I: Overview of tested hyperparameters. The learning rate decay is denoted by the factor the learning rate will be decayed to after 100000 iterations.

	γ	λ	lr	lrd
Block I	25	0.003	0.001	0.1
	50	0.0003	0.0001	0.01
	400	0.00003	0.00001	
Block II	25	0.03	0.0001	0.01
	10	0.003		0.001

Two variables are introduced to measure the performance of the model trained with different sets of hyperparameters for the *prediction maps*. The first variable $\bar{P}_{path}$ is the mean prediction of the model for the class *path* on all grid cells that are labeled in the ground truth as class *path*. It describes the model's capability to identify the ground truth path in the input data. The second variable $\bar{P}_{not\text{-}path}$ quantifies the model's capability to identify the cells correctly that are



(a) $\gamma = 10$, $\lambda = 0.003$, lr $= 0.001$, lrd to 0.001

(b) $\gamma = 50$, $\lambda = 0.0003$, lr $= 0.0001$, lrd to 0.01

(c) $\gamma = 400$, $\lambda = 0.0003$, lr $= 0.001$, lrd to 0.01

Fig. 4: Examples for the *prediction maps* of the proposed model for the three selected operating points defined by their sets of hyperparameters given in the sub-caption of each sub-figure. The *prediction map* is represented by the orange-red heat map, while the past and future path are shown as a red and blue line, respectively.

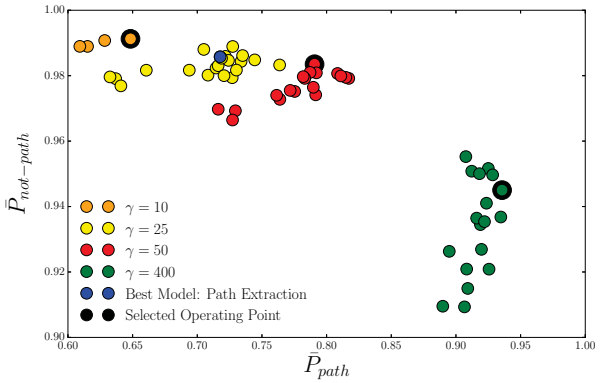


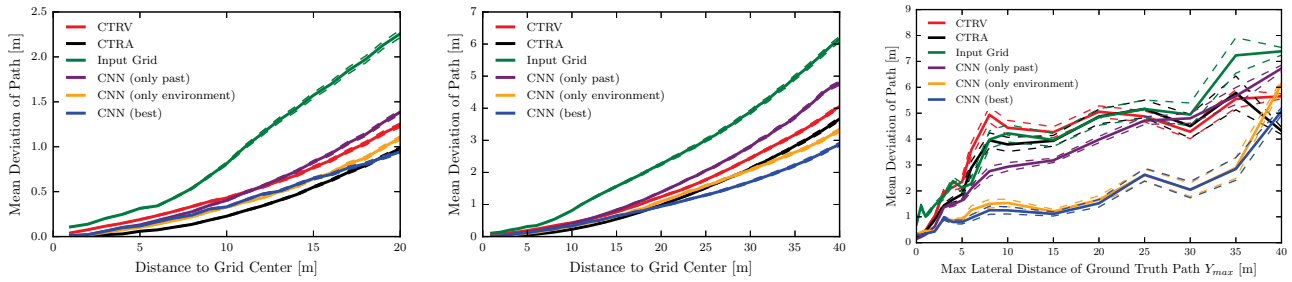
Fig. 5: $\bar{P}_{not\text{-}path}$ vs. $\bar{P}_{path}$ on the validation dataset. Each dot represents the proposed model trained with a different set of hyperparameters color-coded by the scaling factor γ.

not associated with the ground truth path. It is analogously defined by the mean prediction of the model for the class *not-path* on all grid cells that have the ground truth label *not-path*. For a perfect model both variables would be 1. Fig. 5 shows the ROC curve equivalent for this evaluation, by displaying the mean value of $\bar{P}_{not\text{-}path}$ vs. the value of $\bar{P}_{path}$ on the validation dataset for all trained hyperparameter sets color-coded by the hyperparameter γ. This hyperparameter has a large effect on the resulting *prediction maps*, since it weights the contribution of ground truth *path* and *not-path* cells to the loss function. Setting $\gamma = 400$ roughly balances the classes and therefore yields $\bar{P}_{not\text{-}path} \approx \bar{P}_{path}$, while selecting e.g. $\gamma = 10$ yields a large bias for the class *not-path*, resulting in $\bar{P}_{not\text{-}path} > \bar{P}_{path}$, as can be seen in Fig. 5. Furthermore, this figure shows that none of the tested sets of hyperparameters performs exceptionally well if it comes to scoring a large value for both, $\bar{P}_{not\text{-}path}$ and $\bar{P}_{path}$. This means that a trade-off is necessary between high confidence predictions of *path* and *not-path* cells, such that the resulting *prediction map* is optimal for the desired task. Therefore, this paper examines three sets of hyperparameters corresponding to three exemplary chosen operating points of the model. One with a strong suppression of cells being classified incorrectly as part of the path ($\gamma = 10$), one with a high accuracy in the classification of path cells ($\gamma = 400$) and one in between both ($\gamma = 50$). These selected operating points are marked with black dots in Fig. 5. An example of a *prediction map* and the full set of hyperparameters for each selected operating point is given in Fig. 4. It displays a scene, where the Ego-Vehicle is driving towards a T-shaped intersection, with one fork going straight ahead, while the other one turns right. It was selected as it showcases the strengths and weaknesses of the selected operating points: The model with $\gamma = 10$ classifies grid cells within a very slim band for the straight mode as part of the path, while it shies away from confident predictions for the turn right mode. In contrast to this, the model with $\gamma = 400$ classifies cells within a wide band, that covers almost the entire width of the road, for both, the straight and right mode, as part of the path, while the prediction of the model with $\gamma = 50$ is in between those two extremes.

D. Evaluation of the Path Extraction

1) Baseline Models: The extracted paths are tested against three different baseline models. Two of them solely rely on vehicle dynamics. The first assumes a constant turn rate and velocity (*CTRV*) whereas the second assumes a constant turn rate and acceleration (*CTRA*). Within the *CTRA* model, negative acceleration can yield paths that do not reach the edge of the grid, as the path prediction stopped at the zero transition of the velocity. To additionally provide a baseline model that relies on environmental information, we apply the path extraction described in section IV on the environmental input grid (i.e. without an application of the CNN). Hereby, we set the free space probability to 0 for all cells corresponding to the area in the x-y-plane that fulfills $x < -0.5\,|y|$[2]. This is necessary as this approach

[2]See the supporting video for a visualization



(a) Mean deviation vs. distance, zoomed-in. (b) Mean deviation vs. distance. (c) Mean deviation vs. Y_{max}

Fig. 6: Sub-figure (a) and (b) show the mean deviation of the extracted paths vs. the Euclidean distance from the grid center for the first $20\,m$ and the full grid range of $40\,m$, respectively. The mean deviation of the paths vs. $Y_{\max}$ is shown in Sub-figure (c). The dashed line correspond to the $1\,\sigma$ uncertainties originating from the calculation of the mean values.

has otherwise no knowledge of the direction the vehicle is heading to and might predict unreasonable paths laying behind the vehicle.

2) Grid-Based Metric for Comparison of Paths: In order to make a quantitative comparison between the paths predicted by the introduced model and the considered benchmarks it is necessary to define a metric that quantifies the quality of the predicted paths. Within this paper, this is measured by the mean deviation of the predicted path and the ground truth path with the metric $\bar{D}$. It is given by

$$\bar{D} = \frac{1}{n} \sum_{c_p \in C_p} D(c_p, P_{gt}), \tag{4}$$

where C_p denotes the set of all cells from the predicted path and n is the number of cells in C_p. $D(c_p, P_{gt})$ denotes the distance of c_p to the closest cell of the ground truth path P_{gt}.

3) Results: As it is a priori not known which *prediction maps* and therefore which set of hyperparameters yields the best extracted paths, the mean performance on the validation dataset is evaluated with respect to the metric $\bar{D}$ and the best performing set of hyperparameters is selected. This was found to be $lr = 0.0001$, lrd to 0.01, $\gamma = 25$ and $\lambda = 0.003$, which is marked with a blue dot in Fig. 5. The model trained with this set of hyperparameters is from here on denoted by *CNN (best)*. The performance of *CNN (best)* is compared to the baseline models *CTRA*, *CTRV* and *Input Grid*. In a small fraction of samples *CTRA* (6 %) and *CTRV* (0.08 %) predict very short paths (less than 40 grid cells), that only cover the easy to predict vicinity of the Ego-Vehicle and shy away form predicting a path for the more challenging large distances. Due to this, they tend to score much smaller values of $\bar{D}$, compared to the other models, that always predict paths reaching the grid edge, although the predicted paths are of similar quality within the range of the short paths. Therefore, samples with this issue are not considered for the evaluation of the *CTRA* and *CTRV* model. Two additional CNNs, that are identical to the proposed model up to a modification explained below, are trained and evaluated to test the influence of the two input channels of the proposed model. They are modified with respect to the input they receive: *CNN (only path)* only relies on the past path as an input, while *CNN (only environment)* receives only the environmental grid as an input, where the sector lying behind the Ego-Vehicle is blinded as for the baseline *Input Grid*.

TABLE II: Performance with respect to $\bar{D}$ of the proposed model and all baselines averaged over the test dataset. The given uncertainties correspond to the 1σ uncertainties originating from the calculation of the mean.

Model	mean of $\bar{D}$ on test dataset
CTRV	$2.04 \pm 0.03\,m$
CTRA	$1.64 \pm 0.03\,m$
Input Grid	$2.55 \pm 0.04\,m$
CNN (only path)	$1.75 \pm 0.03\,m$
CNN (only environment)	$1.34 \pm 0.03m$
CNN (best)	$1.14 \pm 0.02\,m$

The performance of the proposed models and all considered baseline approaches averaged over the entire test dataset is given in Table II. The proposed model *CNN (best)* outperforms all other models by a large margin. The second best model is *CNN (only environment)* that yields significantly better results than *CNN (only path)*, supporting the hypothesis that the static environment contains valuable information for the prediction of the future Ego-Vehicle path. A closer look at the extracted paths reveals that the poor performance of the non learned environmental aware model *Input Grid* compared to the learned environmental aware models is mostly caused by its poor capability to detect the proper mode in ambiguous scenes. This indicates that a more intelligent detection of the proper mode, as e.g. provided by the learned models is crucial for precise predictions.

Fig. 6 (a) and (b) show the mean deviation of the extracted path predictions and the ground truth path depending on the Euclidean distance of the path element from the current Ego-Vehicle position (center of the grid) for all considered models. Hence, it gives an insight into the accuracy vs. prediction distance. *CTRA* yields better or comparable predictions for small distances up to 20 meter compared to the *CNN (best)* model, while *CNN (best)* outperforms all other models at larger distances. This highlights that the consideration of environmental data is especially valuable for long term predictions, while models that do not consider

environmental data like *CTRA* can perform well for short term predictions, where their model assumptions are most likely fulfilled.

Turn scenarios pose a challenge to the path-predicting models, because turns have to be anticipated correctly and models, relying on measurements of the environment, furthermore have to handle occlusions that are typically present in such scenarios. We compute a feature $Y_{\max}$, which is defined by the maximum lateral distance of the ground truth path to the current position of the Ego-Vehicle, to evaluate the model's performance with respect to turns. Fig. 6 (c) illustrates the average performance of the path predictions vs. $Y_{\max}$, which indicates how well the predictive model handles turns and curvy paths. All considered models perform better than their dataset-wide mean on scenes with a small $Y_{\max}$, which can be seen by comparing Fig. 6(c) with Table II. The learned and environmental aware models significantly outperform all other tested models for scenes with large values of $Y_{\max}$ yielding that they provide more precise predictions in turn scenarios. Although a minimum path length of 40 grid cells is required for *CTRV* and especially *CTRA*, there is still a remnant in the evaluation caused by paths, predicted by *CTRA*, that do not reach the edge of the grid. Due to the typical slowdown of a vehicle approaching a turn, there is relatively large number of short paths predicted *CTRA* in turn scenarios, which yields a relatively small value of $\bar{D}$ for $Y_{\max} = 40m$, as can be seen in Fig. 6 (c).

VI. Conclusion and Outlook

In this work, we proposed a learned model to improve predictions on the future position of the Ego-Vehicle by considering the static vehicle environment. The model operates solely on data measured by on-board sensors and hence does not rely on prior data as e.g. maps. It is capable to handle ambiguous situations like approaching an intersection, by supporting multi-modal predictions in the resulting *prediction maps*. The parametrization of the model allows furthermore to adjust the model's performance, with respect to the accuracy of the predicted path and its capability to correctly identify areas that are not part of the path, to the need of a desired task. Additionally, a method is derived to extract a drivable path from the *prediction maps*. Paths predicted with this method significantly outperform all tested baselines in terms of the mean deviation of the predicted and the ground truth path. The performance gain is especially pronounced in long term predictions and during turns, where the static environment has a large influence on driving behavior. The proposed model has furthermore no intrinsic limitation that restricts it to Ego-Vehicles. It can be directly applied to other vehicles, once the static environment and the past path of this vehicle is known.

One promising aspect for future work is to incorporate features, like the usage of turn signals, which contain information that helps to resolve the ambiguity between modes. Conditioning the model on such features might enable a more timely prediction of the correct modes at e.g. intersections. Besides adding such features, it might also be beneficial to exploit the sequential nature of the underlying problem by modifying the model's architecture such that its predictions are based on a whole sequence of environmental measurements and not solely on the most recent one.

Another intriguing aspect is to extend the prediction model to the temporal dimension and hence predict trajectories instead of paths. For this, it is necessary to extend the input space to features, like dynamic objects, speed limits, or right of way, as they strongly influence vehicle's trajectories.

References

[1] R. Schubert, E. Richter, and G. Wanielik, "Comparison and evaluation of advanced motion models for vehicle tracking," in *IEEE 11th International Conference on Information Fusion*, 2008.

[2] S. Ammoun and F. Nashashibi, "Real time trajectory prediction for collision risk estimation between vehicles," in *IEEE 5th International Conference on Intelligent Computer Communication and Processing*, 2009.

[3] A. Tamke, T. Dang, and G. Breuel, "A flexible method for criticality assessment in driver assistance systems," in *IEEE Intelligent Vehicles Symposium (IV)*, 2011.

[4] J. Wiest, M. Höffken, U. Kreßel, and K. Dietmayer, "Probabilistic trajectory prediction with gaussian mixture models," in *IEEE Intelligent Vehicles Symposium (IV)*, 2012.

[5] F. Large, D. A. V. Govea, T. Fraichard, and C. Laugier, "Avoiding cars and pedestrians using v-obstacles and motion prediction," in *Proc. of the IEEE Intelligent Vehicle Symp.*, 2004.

[6] C. Hermes, C. Wohler, K. Schenk, and F. Kummert, "Long-term vehicle motion prediction," in *IEEE Intelligent Vehicles Symposium*, 2009.

[7] S. Lefèvre, C. Laugier, and J. Ibañez-Guzmán, "Exploiting map information for driver intention estimation at road intersections," in *IEEE Intelligent Vehicles Symposium (IV)*, 2011.

[8] A. Houenou, P. Bonnifait, V. Cherfaoui, and W. Yao, "Vehicle trajectory prediction based on motion model and maneuver recognition," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[9] A. Polychronopoulos, M. Tsogas, A. J. Amditis, and L. Andreone, "Sensor fusion for predicting vehicles' path for collision avoidance systems," *IEEE Transactions on Intelligent Transportation Systems*, 2007.

[10] D. Barnes, W. Maddern, and I. Posner, "Find your own way: Weakly-supervised segmentation of path proposals for urban autonomy," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[11] L. Caltagirone, M. Bellone, L. Svensson, and M. Wahde, "Lidar-based driving path generation using fully convolutional neural networks," in *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2017.

[12] 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.

[13] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015.

[14] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML)*, 2010.

[15] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[16] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010.

[17] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *3rd International Conference for Learning Representations*, 2014.

[18] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.