for this purpose was introduced in 2020 with VectorNet [23]. Thus, VectorNet is a solid baseline and ideally suitable for our application. The problem is flipped: In typical motion prediction tasks, AVs predict surrounding objects' trajectories. In testing, the trajectories of surrounding objects are (partially) known, but the AVs' behavior has to be predicted.

Scenario embeddings do not cover all properties of instantiated scenarios. The relevant properties may have to be determined by feature selection. Remaining properties can be accounted for by probabilistic motion prediction models.

IV. Experiments

Below, we describe our investigation aimed at addressing three main questions: Can VectorNet predict Ego trajectories...

- for individual functional scenarios it is trained on?
- for multiple functional scenarios it is trained on?
- for functional scenarios not seen during training?

A. Generation of the Scenario Embeddings and Trajectories

To enable a comparison to regression metamodels, we generate vectorized scenario embeddings for the three scenarios in Fig. 2 parametrically but introduce random variations making the scenarios more complex and realistic (see Fig. 4).

The center of the lanes follows a polynomial of 3rd degree with $x \in [-55\,\mathrm{m}, 55\,\mathrm{m}]$; where vectors describe the lanes (at $-55\,\mathrm{m} + \frac{110\,\mathrm{m}}{24} \cdot i \;\forall\; i \in \mathbb{N}_0^{\leq 24}$), the y-coordinate of the lane's center is shifted uniformly by $\mathcal{U}(-0.5\,\mathrm{m}, 0.5\,\mathrm{m})$. Accordingly, the lane width is $3.5\,\mathrm{m} + \mathcal{U}(-0.3\,\mathrm{m}, 0.3\,\mathrm{m})$. Ego and Co use a lane keeping controller; the resulting vehicle orientations are varied randomly with $\mathcal{U}(-2.9°, 2.9°)$. The Co's velocity follows the description in Fig. 2 and is varied with $\mathcal{U}(-0.1\,\frac{\mathrm{m}}{\mathrm{s}}, 0.1\,\frac{\mathrm{m}}{\mathrm{s}})$. Using a P-controller, the Ego tries to ensure a time gap of $2\,\mathrm{s}$ to the Co. Concrete scenarios are sampled uniformly from the ranges given in Table I. The time step width is $0.2\,\mathrm{s}$, and the scenarios are simulated for $5\,\mathrm{s}$.

B. Training of the Motion Prediction Model VectorNet

We build upon a publically available implementation of VectorNet [9]. VectorNet is supplied with the lanes, the Co's trajectory, and the initial vector of the Ego (which holds information about the position, orientation, and velocity). It has to predict all following vectors of the Ego. Available data are split into 90% training and 10% validation data.

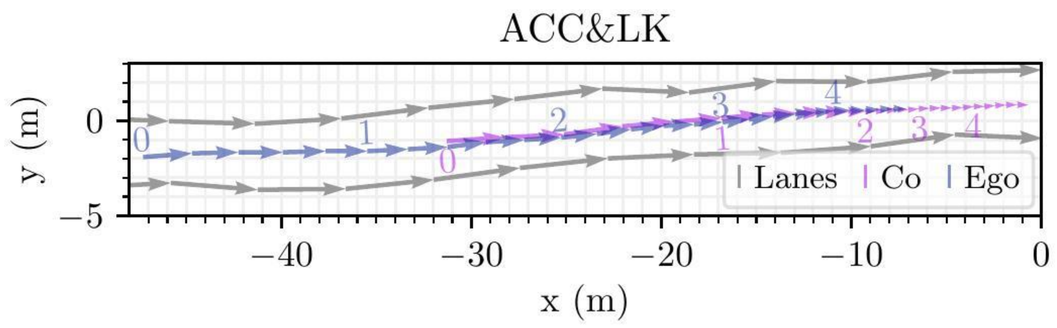

Fig. 4. Data of the scenario ACC&LK. Each vector is defined by its start and end points, object type, object ID, and timestamp. The numbers next to the vectors represent their timestamps in seconds; lanes' timestamps are 0. Note that vectors can contain more features, e.g., bounding boxes or object colors.

C. Evaluation Procedure and Criteria

To answer the research questions, VectorNet is trained with different mixes of the three scenarios' data. N_{LK}, N_{ACC}, and $N_{\mathrm{ACC\&LK}}$ denote the number of samples of the respective scenario VectorNet is supplied with. $\mathrm{ADE}_{\mathrm{LK}}$, $\mathrm{ADE}_{\mathrm{ACC}}$, and $\mathrm{ADE}_{\mathrm{ACC\&LK}}$ denote the average displacement error (ADE) with respect to the Ego's trajectory. The ADEs are calculated based on a test set with a size of 10k per functional scenario.

As shown in Fig. 1, we also use VectorNet's predictions to calculate evaluation metrics and compare these to predictions of regression metamodels, which predict the evaluation metrics based on the respective inputs of the scenarios in Table I.

V. Results

Below, we assess VectorNet's predictive performance based on the predicted Ego trajectories and evaluation metrics.

A. Assessment Based on Predicted Ego Trajectories

Rows 1 to 3 of Table II indicate that trained with individual functional scenarios' data, VectorNet achieves good ADEs for the functional scenario it is trained on. Rows 4 to 7 show that combining data from two functional scenarios is possible; however, for LK and ACC&LK, 3k samples are required. Combining data from all three scenarios (row 8), the predictive performance is similar to that of the separately trained models (rows 1 to 3); Fig. 5 visualizes this model's predictions.

The more challenging task is the generalization to functional scenarios not seen during training. Row 3 shows that trained on ACC&LK, a low ADE for ACC is achieved. This is expected

TABLE I
Logical Scenarios' Inputs and Ranges

Input	Min	Max	Unit	Explanation
a_0	-1	1	-	0^{th} polynomial coefficient
a_1	-0.1	0.1	-	1^{st} polynomial coefficient
a_2	-0.01	0.01	-	2^{nd} polynomial coefficient
a_3	-0.001	0.001	-	3^{rd} polynomial coefficient
v_{Ego}	8	16	$\frac{\mathrm{m}}{\mathrm{s}}$	Ego's initial velocity
x_{Co}	$-50 + v_{\mathrm{Ego}}$	$-50 + 2v_{\mathrm{Ego}}$	m	Co's initial x-coordinate
v_{Co}	$v_{\mathrm{Ego}} - 4$	$v_{\mathrm{Ego}} + 4$	$\frac{\mathrm{m}}{\mathrm{s}}$	Co's initial velocity
$t_{\mathrm{v\ Co}}$	0	3	s	duration of constant velocity
a_{Co}	-8	-1	$\frac{\mathrm{m}}{\mathrm{s}^2}$	Co's acceleration
$t_{\mathrm{a\ Co}}$	1	3	s	duration of acceleration

TABLE II
Average Displacement Errors (ADE) of Pred. Ego Trajectories

Row	N_{ACC}	N_{LK}	$N_{\mathrm{ACC\&LK}}$	$\mathrm{ADE}_{\mathrm{ACC}}$	$\mathrm{ADE}_{\mathrm{LK}}$	$\mathrm{ADE}_{\mathrm{ACC\&LK}}$
1	2000	0	0	**0.37** m	12.47 m	8.96 m
2	0	2000	0	2.40 m	**0.40** m	6.48 m
3	0	0	2000	0.43 m	14.86 m	0.52 m
4	2000	2000	0	0.39 m	0.41 m	4.69 m
5	0	2000	2000	1.58 m	0.97 m	1.73 m
6	0	3000	3000	0.45 m	0.46 m	0.54 m
7	2000	0	2000	**0.37** m	9.18 m	**0.47** m
8	2000	2000	2000	0.40 m	0.42 m	0.60 m
9	2000	2000	200	0.41 m	0.42 m	0.83 m
10	0	0	200	0.88 m	9.95 m	1.13 m