IEEE Robotics and Automation Letters (RAL) paper presented at the
2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Deep Context Maps: Agent Trajectory Prediction using Location-specific Latent Maps

Igor Gilitschenski[1], Guy Rosman[2], Arjun Gupta[1], Sertac Karaman[1], Daniela Rus[1]

Abstract— **In this paper, we propose a novel approach for agent motion prediction in cluttered environments. One of the main challenges in predicting agent motion is accounting for location and context-specific information. Our main contribution is the concept of learning context maps to improve the prediction task. Context maps are a set of location-specific latent maps that are trained alongside the predictor. Thus, the proposed maps are capable of capturing location context beyond visual context cues (e.g. usual average speeds and typical trajectories) or predefined map primitives (such as lanes and stop lines). We pose context map learning as a multi-task training problem and describe our map model and its incorporation into a state-of-the-art trajectory predictor. In extensive experiments, it is shown that use of learned maps can significantly improve predictor accuracy. Furthermore, the performance can be additionally boosted by providing partial knowledge of map semantics.**

I. INTRODUCTION

Trajectory prediction of diverse agents in dynamic environments is a key challenge towards unlocking the full potential of autonomous mobile robots. Particularly in safety critical settings such as autonomous driving, obtaining reliable predictions of surrounding agents is a necessary functionality for robust operation at speeds comparable to human-driven vehicles. Predicting agent trajectories, such as pedestrian motions, is an inherently challenging task: First, in crowded environments, pedestrian behavior is highly dependent on social interactions. Second, prediction systems have to account for potential rapid changes in behaviour (e.g. children unexpectedly running on the road). Finally, the decision making process involves and depends on a high variety of factors including the surrounding environment and the complex interactions with it.

Several deep-learning based approaches have recently incorporated context to improve prediction accuracy [1], [2], [3]. This was usually achieved by providing a top-down view of the scenery to the prediction network [3], or by adding information (e.g. about lanes) from a map of the area [4]. This improves prediction performance but fails to account for information that is not available through visual cues or the content of a map. Furthermore, existing map-based prediction approaches usually rely on a map creation process that requires additional engineering and reasoning about useful features for prediction. To the best of our knowledge, there is no prediction approach that implicitly learns a task-specific latent map.

Manuscript received: February, 25, 2020; Revised May, 14, 2020; Accepted June, 4, 2020.

This paper was recommended for publication by Editor Tamim Asfour upon evaluation of the Associate Editor and Reviewers' comments. The work has been supported by the Toyota Research Institute (TRI). It, however, reflects solely the opinions and conclusions of its authors and not TRI or any other Toyota entity. The support is gratefully acknowledged.

[1]Igor Gilitschenski and Daniela Rus are with Computer Science and Artificial Intelligence Lab, Massachusetts Institute of Technology, USA igilitschenski@mit.edu, rus@csail.mit.edu

[2]Guy Rosman is with Toyota Research Institute, USA guy.rosman@tri.global

[3]Arjun Gupta and Sertac Karaman are with Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, USA argupta@mit.edu, sertac@mit.edu

Digital Object Identifier (DOI): see top of this page.

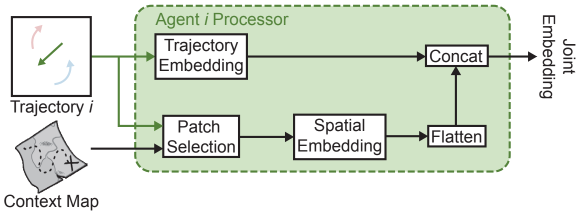


Fig. 1: **Map Embedding.** Latent maps can be used for learning location-specific information. For the trajectory forecasting task, we preprocess the latent map and the trajectory resulting in a joint trajectory-map embedding.

We argue that prediction performance can be improved by revising the way predictors and maps interact. Typical local behavior is an important cue in predicting behaviors and affordances. This is strongly evident in autonomous vehicle fleets that regularly traverse the same roads. Maps should therefore extend beyond the content of a top-down image or an HD map with fixed semantics. They should provide predictors with location-specific context in a structured way. We propose learning a location-specific joint representation (called context map) that explains top-down views, semantic primitives, and the behavior of agents operating in the scene.

For instance, the ever-growing amount of raw trajectory data contains a lot of information about local norms. Trajectories also implicitly encode phenomena such as common paths, potential obstacles, and traffic flow. Thus, they could be a rich source of non-visual information. However, it is an open problem how to incorporate such information into a context representation for the prediction task. A related challenge is the decomposition of location-specific and location-agnostic learned structures within the predictor's architecture.

In this paper, we address these challenges by proposing an approach for neural network-based prediction that incorporates context maps, a learned location-specific memory. Instead of presuming a predetermined structure, we train the maps as latent entities that can not only explain visual

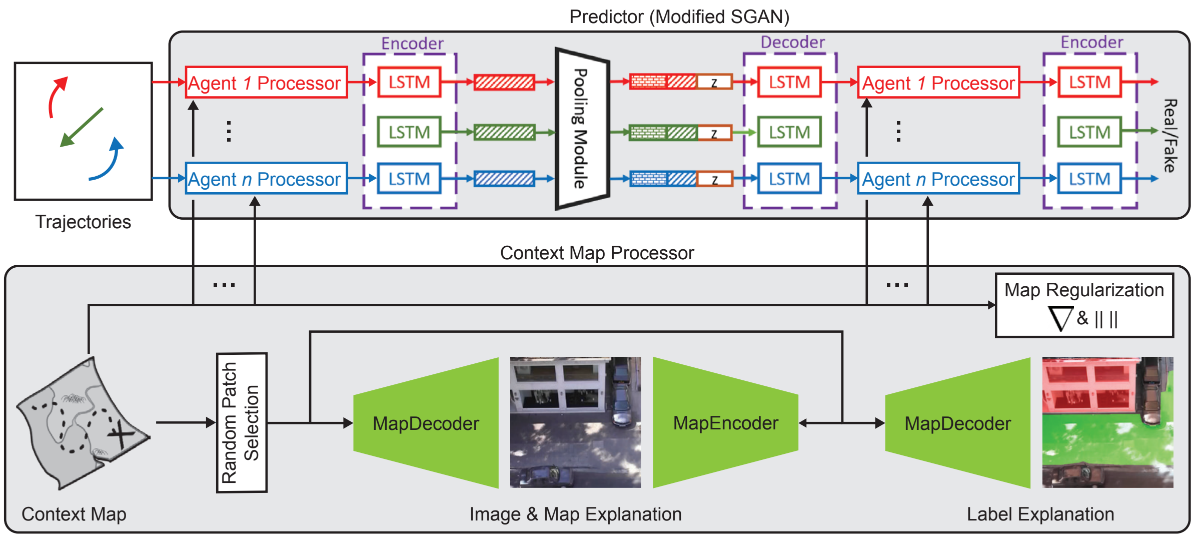


Fig. 2: **Architecture Overview.** The proposed overall architecture for learning context maps jointly with trajectory prediction. For the prediction network, we use a modified variant of the Social GAN architecture ([5, Fig.2]) that introduces the context map after the trajectory embedding [5, Eqns. 2 & 4]. Additional supervision for learning the context map is provided image explanation, map explanation, and label explanation losses on randomly sampled map patches and the corresponding image patches. Finally, the map is regularized using a norm penalization loss, and a gradient norm penalization loss to enforce sparsity of the map gradient.

features of the image, but also non-visual features for the prediction task. The trained maps are implemented as a set of location-specific biases that are injected into the prediction network. Additional auxiliary loss terms based on reconstruction, partial semantic annotations, and gradient sparsity provide support to guide the map-learning process. At test time, the network uses current locations of the agents and the learned maps as input for prediction. To the best of our knowledge, this is the first work focusing explicitly on latent map learning for improving the prediction task. Overall, our contributions can be summarized as follows:

- We develop a model for utilizing latent maps to explain trajectories of road agents and integrate them as part of a state-of-the-art trajectory prediction network.
- We show how we learn the maps from raw aerial imagery as well as the observed motion patterns and partial semantic labels.
- We demonstrate how the learned maps allow us to better predict agents' motion and outperform baseline approaches for the prediction task. We show this on standard benchmark datasets and probe the performance contribution of the maps and additional semantic labels.

II. RELATED WORK

Trajectory modeling and prediction has been covered in an extensive and diverse body of work. Early works consider prediction in the context of tracking [6] or social interaction modelling [7]. Recent advances in deployment of autonomous vehicles have sparked renewed interest in the prediction task (see [8], [9], [10] for recent surveys). A main challenge remains the incorporation of environmental context.

Social-Context Modelling One big line of work, inspired by [7], considers improving prediction by properly modelling social context and group dynamics. In [11], future trajectories of all interacting agents are modeled by learning social interactions from real data using a Gaussian process model. An interaction-aware prediction network is used in [12] for safely crossing an intersection. In [13], static obstacles and surrounding pedestrians are explicitly modeled for improving the forecasting task while [14] proposes an approach that uses graphs for scene representation. Social Pooling modules are proposed in *Social LSTM* [15] and *Social GAN* [5]. They allow a deep learning-based predictor to jointly reason about multiple agent trajectories. In contrast to these approaches, the present work considers the orthogonal problem of semantic context modelling and can be combined with modelling social context as we demonstrate by integrating Context Maps with *Social GAN*. We show that proper modelling of semantic context has a stronger impact on prediction accuracy than social pooling.

Semantic Context Modelling. Several recent approaches consider broader semantic context information for trajectory prediction. In [16], previously observed motion patterns are used to estimate a probability distribution as motion model and [17] estimates circular distributions at different locations which are combined into a smooth path prediction. An existing map with annotated traffic lanes and centerlines is used for prediction in [18] based on a Kalman Filter framework for predicting vehicle motion. The work [19] proposes to extract patch descriptors that encode the probability of moving to adjacent patches and then uses a Dynamic Bayesian Network for scene prediction. In [20], context features from the

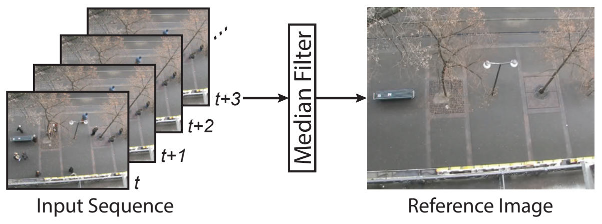


Fig. 3: **Reference Images.** We create reference images of the scenes by applying a median filter. This results in the removal of most dynamic objects and supports scene reconstruction.

environment (such as traffic light status and distance to curbside) are incorporated into a Gaussian Process-based predictor. Several deep learning-based approaches use visual context cues for improving on the prediction task [1], [21], [2], [22], [23], [3] by taking an image of the scenery as an additional input to a network. Different map representations for prediction in scenarios with static trackers are discussed in [24], [25], [26], [27]. An explicit destination network is used in [28] to model a grid of potential destinations for subsequent trajectory prediction. An implicit consideration of semantic information is achieved in networks that combine multiple tasks such as [29], where a single convolutional network is used to combine detection, tracking, and motion prediction. In contrast to these works, our work extracts location-specific semantic information during training and stores it in a learned latent representation. We also use different information sources to inform that representation.

Latent Representation Learning. Our work draws some inspiration from several approaches that encode learned map representations. An early approach of map learning was presented in [30], where a Gaussian Process is used for occupancy mapping without a priori discretization of the world into grid cells. In [31], a differentiable mapper is used to create a multiscale belief of the world in the agent's coordinate frame. Other recent neural mapping approaches involve [32], [33], [34]. While these works mostly focus on localization and navigation, we use a latent location-specific memory to inform the prediction task.

The most similar work to ours is the predictor proposed in [35]. This approach uses a displacement volume as a network input and proposes a location-specific bias map of the size of that volume. In contrast to this work, we consider a model capable of simultaneously handling multiple scenes and multiple map layers. We also propose a location-specific training methodology and additional semantic supervision to improve latent map quality.

III. CONTEXT MAP LEARNING FOR PREDICTION

The goal of the proposed approach is to capture the ability of humans to account for environment and location-specific habits and norms. We model this information by a set of location-specific maps that are learned during predictor training.

More formally, our goal is to predict a set of agent trajectories $\hat{\mathbf{Y}} = \{Y_1, ... Y_N\}$ at a place $l \in \mathcal{P}$ (with $\mathcal{P}$ denoting the set of all considered places) from past temporally overlapping trajectories $\mathbf{X} = \{X_1, \ldots, X_N\}$ and a learned location-specific map M_l, i.e.

$$\hat{\mathbf{Y}} = f(\mathbf{X}, M_l) \ .$$

such that $\hat{\mathbf{Y}}$ approximates the ground truth trajectories, $\mathbf{Y}$, as closely as possible. The trajectories are represented as sequences $X_i = \{\mathbf{x}_{i,t} \in \mathbb{R}^2 \,|\, t = 1, \ldots, O\}$ and $Y_i = \{\mathbf{y}_{i,t} \in \mathbb{R}^2 \,|\, t = 1, \ldots, P\}$ with observation horizon O and prediction horizon P. The neural network representing f is trained together with the context maps M_l. The maps are stored as tensors of size $H_l \times W_l \times F_{map}$ with H_l, W_l denoting the reference image dimensions and F_{map} the map feature dimension. In our case, reference images are usually top-down views of the environment obtained from the video data of the considered datasets via median filtering as shown in Fig. 3. The resulting reference images are visualized in Fig. 4.

In addition to trajectory losses during predictor training (Sec. III-A), we provide weak supervisory information to obtain meaningful maps and ensure convergence. We train the maps to reconstruct the reference image (Sec III-C), to encode information about environment semantics without providing full labels on the entire reference image (Sec III-D), and add a gradient based penalty term to support map smoothness (Sec III-E).

A. Predictor Integration

Recently several generative trajectory prediction approaches based on Generative Adversarial Networks (GANs) [36] have been proposed demonstrating the capability for covering a variety of different plausible trajectories [5], [22], [2]. Motivated by these results and in order to prove usefulness of maps even with elaborate predictors, we integrated our map learning approach with Social GAN (S-GAN) [5] as visualized in Fig. 2. The concept of context map learning is applicable to most neural network based predictors. We used S-GAN due to the free availability of its implementation allowing for a fair baseline comparison.

The S-GAN generator network creates trajectory predictions through a Long Short-Term Memory (LSTM) [37] network which is broadly used by several of the above-mentioned trajectory prediction models. Traditionally, for trajectory prediction, the LSTM network takes in a sequence of agent coordinates, encodes them into a state vector, and a separate predictor network converts the state vector to the future agent location. The S-GAN network simultaneously processes the trajectories of all the pedestrians in a given consecutive sequence of video frames and then "pools" the resulting state vectors of the separate LSTMs before making a prediction. The pooling mechanism serves for modeling social interactions. More formally, for each trajectory X_i, the S-GAN LSTM cell follows the following recurrence:

$$\mathbf{e}_c = \mathrm{MLP}(\mathbf{x}_{i,t}) \ , \quad \mathbf{h}_t = \mathrm{LSTM}(\mathbf{h}_{t-1}, \mathbf{e}_c) \ , \qquad (1)$$

where MLP denotes a multi layer perceptron meant to encode the coordinates of the agent, and $\mathbf{h}_t$ is the hidden state of the LSTM at time t. This computation is carried our for each trajectory in $\mathbf{X}$ individually, however for simplicity of notation, we do not carry the index of the trajectory as it is clear for the context.

As the model is a GAN, it also includes a discriminator network which scores the trajectory produced by the generator. This network is only used during training to improve the generator and not part of the trajectory prediction network at inference time.

We integrate context maps with the S-GAN model by providing an additional input during the prediction phase to the LSTM. We add a Convolutional Neural Network (CNN) that takes in a patch of the context map for the given scene at the current coordinate location and provides a processed form to the first LSTM cell in the generator. This additional input changes the recurrence in (1) to

$$\begin{aligned} \mathbf{e}_m &= \mathrm{MapDecoder}_\mathrm{P}(C_{i,t})\,, \quad & \mathbf{e}_t &= \mathrm{MLP}(\mathbf{x}_{i,t})\,, \\ \mathbf{e}_c &= \mathrm{Concat}(\mathbf{e}_t, \mathbf{e}_m)\,, & \mathbf{h}_t &= \mathrm{LSTM}(\mathbf{h}_{t-1}, \mathbf{e}_c)\,, \end{aligned} \tag{2}$$

where $\mathrm{MapDecoder}_\mathrm{P}(\cdot)$ is a Convolutional Neural Network creating a spatial embedding of the map and $C_{i,t}$ is a patch of the context map around the location $\mathbf{x}_{i,t}$. This process is visualized in Fig. 1. To make a prediction on the future location of the agent, we pass the most recent LSTM state vector to a fully connected decoder network which outputs the future position of the agent (analogous to [5, eq. (4)]).

At training time, the generator involves two loss terms. In addition to the usual discriminator score $\mathcal{L}_{score}$, S-GAN uses a L2 loss term between the predicted trajectory and the true trajectory

$$\mathcal{L}_\mathrm{traj}(\hat{\mathbf{Y}}, \mathbf{Y}) = \sum_{i=1}^{N} \sum_{t=1}^{P} ||\hat{\mathbf{y}}_{i,t} - \mathbf{y}_{i,t}||^2 \ . \tag{3}$$

In our modified version, the discriminator also gets access to the maps using the same augmentation process as for the observed trajectories. However, in contrast to generator training, the latent map is not modified during discriminator training iterations.

B. Map Patch Selection

One of the conceptual choices for the predictor was deciding between using global context of the entire scene (as is done e.g. in [22]) or a local context notion. We decided for the latter to simplify the learning task. To ensure local use of context, the network is implemented such that during each training and inference step only those parts of the map M_l are used that correspond to locations in the current batch or, for training only, to patches selected for auxiliary tasks. This requires a patch selection mechanism which is simply extracting subtensors of the map around a given point $\mathbf{x}$ of size $H_{patch} \times W_{patch} \times F_{map}$ denoted as

$$C = \mathrm{PatchSelection}(M_l, \mathbf{x}) \ .$$

Thus, $C_{i,t}$ above is obtained as $C_{i,t} = \mathrm{PatchSelection}(M_l, \mathbf{x}_{i,t})$.

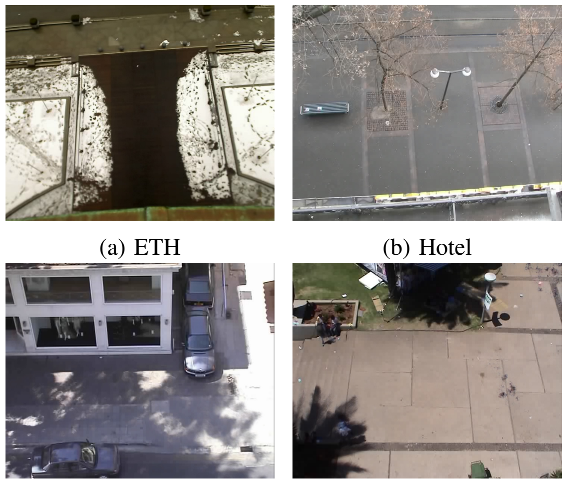


Fig. 4: **Reference Images.** The reference images that are used as supervisory reconstruction labels for our networks are generated by applying a temporal median filter [38] to the video data in the datasets.

C. Image Explanation

Unless the size of the training set for a specific environment is very large, the trajectories will usually not cover all walkable areas and, on their own, do not provide sufficient information about other objects in the environment. To help the network better learn key features of the environment, we introduce an image explanation mechanism as well as a map explanation mechanism. We enforce this constraint by including decoder and encoder modules according to

$$\begin{aligned} \mathcal{L}_\mathrm{image}(l) = &||I_l - \mathrm{MapDecoder}_\mathrm{R}(M_l)|| \\ &+ ||M_l - \mathrm{MapEncoder}_\mathrm{R}(I_l)|| \ . \end{aligned} \tag{4}$$

We note that this methodology can potentially allow to use this prediction approach in unseen scenes.

D. Semantic Label Reconstruction

We expect the latent map to decode semantic labels of the scene where the annotation is present. To that end, we add another module to decode the context map into semantic class labels represented as a matrix $L_{l,i} \in \mathbb{R}^{H_l \times W_l}$ for each location l and label type i. For positive and negative examples we set the values of $L_{l,i}$ to 1 and -1 respectively and leave it at 0 if there is no label information. The label reconstruction loss is formulated as the difference between the hand-annotated semantic labels and the decoded annotation

$$\mathcal{L}_\mathrm{labels}(l) = \sum_{i \in \mathcal{T}} (L_{l,i} - B_{l,i} \circ \mathrm{MapDecoder}_S(M_l)_i)^2 \ ,$$

where $\mathcal{T}$ denotes the set representing label types, $\circ$ is the Hadamard product and $B_{l,i}$ is a bitmask ensuring that no loss is incurred for areas without any label (i.e. it is 0 wherever $L_{l,i}$ is 0 and 1 otherwise).

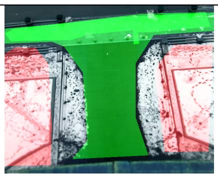
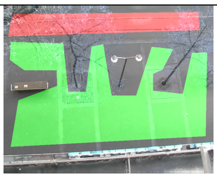
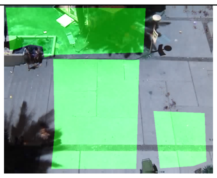
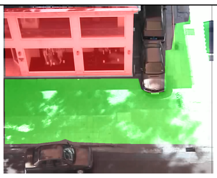
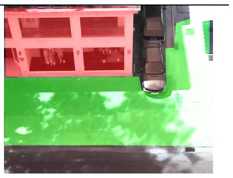
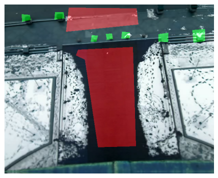
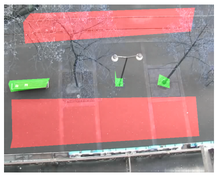
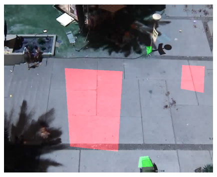
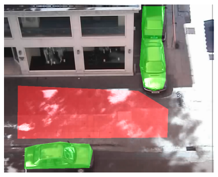
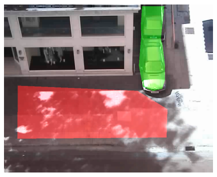
Category	ETH	HOTEL	STUDENTS3	ZARA1	ZARA2
Walkable					
Obstacle					

TABLE I: **Annotated labels.** The labels are given in terms of positive examples (green) and negative examples (red) encoded as 1 and -1 respectively. They do not cover the full corresponding semantic area in every reference image. We demonstrate that the information encoded in those labels can still be used to learn segmentation and support prediction in unlabeled areas.

E. Sparsity

In order to ensure that the map is as simple as possible, we add a sparsity prior on the gradient of the latent map, $\mathcal{L}_{\text{sparsity}}(l) = ||\nabla M_l||_1$. This is implemented using a finite differences approach computed by applying a convolution with the two predefined kernels

$$\begin{bmatrix} 0 & 0 & 0 \\ \varepsilon & -\varepsilon & 0 \\ 0 & 0 & 0 \end{bmatrix} \text{ and } \begin{bmatrix} 0 & \varepsilon & 0 \\ 0 & -\varepsilon & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

to the latent map. Then, we compute the norm by treating the resulting output as a vector. We refer the reader to [39] and references therein for further details about sparse representations, and note that other image priors could have been used as well.

F. Training

For a given scene, we initialize the context map and network with random weights picked from a Gaussian distribution. We alternate training between the trajectory generator network and the discriminator network. To further balance the training, the auxiliary map learning losses are additionally restricted to a set of randomly selected patches rather than the entire latent map.

1) Generator Step: To train the generator network, we initially do a forward pass by first selecting sections of the context map dependent on the scene and pedestrian coordinates in the batch. We input the coordinates and map sections into the network to predict the trajectories. We compute several losses with respect to the context map and the resulting trajectories to train the network.

During the training phase, we not only train the weights of the encoder as well as the Social-GAN modules but also the selected patches of the context map. Thus, the training process enforces location-specific information in the context map. The selected losses make sure that the information stored in the map informs the trajectory prediction network to make more accurate predictions and do so concisely in a way that captures the key components of the reference image.

The total loss for the generator is thus obtained as

$$\begin{aligned} \mathcal{L}_{\text{G}}(\hat{\mathbf{Y}}, \mathbf{Y}, l) = {} & w_1 \cdot \mathcal{L}_{\text{image}}(l) + w_2 \cdot \mathcal{L}_{\text{labels}}(l) \\ & + w_3 \cdot \mathcal{L}_{\text{sparsity}}(l) + w_4 \cdot \mathcal{L}_{\text{score}}(\hat{\mathbf{Y}}) \\ & + w_5 \cdot \mathcal{L}_{\text{traj}}(\hat{\mathbf{Y}}, \mathbf{Y}) \ . \end{aligned}$$

2) Discriminator Step: The descriminator is trained to provide $\mathcal{L}_{\text{score}}$ in the same way as in Social GAN. We first pass a partial pedestrian trajectory to the trajectory generator to get a predicted full trajectory. We then pass the predicted trajectory and true trajectory to the discriminator network to obtain scores for the two trajectories. While we did not modify the loss of the discriminator, we adapted the discriminator's first LSTM cell in the same way as in the generator. However, the discriminator does not have an own map encoder module and gets merely read-only access to the generator's maps, i.e. they are not changed during discriminator training. Letting the discriminator see the maps without modifying them allows for better convergence while not violating the GAN's theoretical equilibrium properties.

IV. EVALUATION

In our evaluation, we aim to capture the effects of using context maps on trajectory prediction accuracy. By explicitly discussing dataset imbalances and different scales, we demonstrate the utility of learned maps in the presence of a diverse set of data. In what follows we first introduce the baselines (Sec. IV-A) and datasets (Sec. IV-B), followed by the implementation details (Sec. IV-C) of our evaluation, and finally a discussion of our results (Sec. IV-D).

A. Baselines

We evaluate the proposed approach along the following baselines including several variants of our own model to understand the individual contribution of each component:

1) Linear: A simple Kalman filter (we used pykalman 0.9.5) running a constant acceleration model where the initial state covariance, the model covariance, and the observation covariance were estimated using an expectation maximization method for each trajectory individually.

Sequence	Linear	S-GAN	S-GAN-P	Ours	Ours no pooling	Ours no labels
ETH	16.33 / 35.09	38.25 / 67.35	44.62 / 81.96	17.64 / 34.20	**14.63 / 26.83**	15.77 / 28.89
HOTEL	20.81 / 44.68	23.52 / 39.57	25.32 / 42.41	19.12 / 34.62	**18.79 / 33.77**	20.90 / 38.20
ZARA1	21.44 / 49.16	28.60 / 50.08	30.87 / 53.45	**17.79 / 34.54**	17.99 / 35.69	24.37 / 48.63
ZARA2	14.64 / 34.32	19.48 / 34.56	19.21 / 33.83	13.43 / **26.20**	**13.32** / 26.40	15.88 / 31.09
STUDENTS3	30.86 / 71.38	28.95 / 53.87	29.55 / 54.81	**22.02 / 43.84**	22.75 / 45.94	23.13 / 45.82
Average	20.81 / 46.93	27.76 / 49.09	29.91 / 53.29	18.00 / 34.68	**17.50 / 33.72**	20.01 / 38.53

TABLE II: **Prediction Results** given in terms of ADE (left) and FDE (right) in pixels for each sequence individually and a (non-weighted) average. Use of context maps outperforms approaches that purely predict from trajectory data. Particularly datasets underrepresented in the training data (in our case ETH) stand to benefit from the use of location-specific maps.

2) S-GAN-P: We compare our approach against the full S-GAN model including their proposed pooling module. We use the S-GAN predictor (with the modifications outlined above) in our training pipeline and thus, this serves at the same time as an ablation study for the use of context maps.

3) S-GAN: This baseline is basically the same as S-GAN-P with the only difference being the removal of the pooling module. That is, we train a simple LSTM-based GAN for trajectory prediction.

4) Ours: Our full model involving all loss terms described above.

5) Ours no pooling: Our full model without the pooling module in the generator to evaluate the relative contribution of pooling.

6) Ours w.o. labels: Our full model without semantic labels in order to evaluate if weak semantic supervision helps prediction.

B. Datasets

The evaluation of our model requires datasets with trajectory data of multiple trajectories in the same location and a corresponding reference image of the scenery. Unfortunately, no such autonomous vehicle data is publicly available yet. Thus, we use datasets based on a static tracker and evaluate our work on the ETH and UCY standard benchmark datasets.

ETH Dataset [40] The ETH dataset, also known as BIWI Walking Pedestrian dataset, is a collection of two sequences (ETH and HOTEL) recorded around ETH Zurich. For both sequences, it contains pedestrian positions and velocities in meters and a video recording. Furthermore, it provides homography matrices for transforming the data into image coordinates. For dataset compatibility and to capture if the maps help with different scales, we use pixel coordinates for this dataset.

UCY Dataset [41]. The UCY "Crowds-by-Example" dataset contains several sequences with annotated pixel location trajectories. Not all of these sequences also have a corresponding video recording or the viewpoint of the recording, is not suitable. Thus, we make only use of the ZARA1, ZARA2, and STUDENTS3 sequences.

We process each dataset by stepping through sequences of frames and taking subsequences of size 18 (with an observation sequence length of $O = 10$ and an prediction sequence length of $P = 8$). We only include pedestrians that appear in the full sequence of sampled frames. Partial pedestrian sequences are dropped. Since we use a sliding window approach, taking multiple (only partially overlapping) subsequences from each trajectory, we significantly increase the size and variety of our dataset.

UCY is annotated at a rate of ten times that of ETH, so we only take every tenth frame of UCY to ensure that there are no large discrepancies in temporal scaling between the generated sequences from the two datasets. At the same time, we use each datapoint in both datasets as a potential starting point for a sequence. This data augmentation measure creates several partially overlapping sequences with more datapoints from UCY than ETH. We do not compensate for this in order to evaluate our hypothesis that context maps can better account for imbalance by properly learning location-specific information.

Once the datasets are processed, we split 10% of the data in each scene into a validation set and 30% of the data into a test set. The remaining 60% are used for training. It is important to note that these percentages are taken from each scene, not from the overall pool. This ensures that there is a good representation of each scene in each of the splits. Furthermore, the splits were made on a temporal basis making sure that no two trajectories from the same point in time end up in different splits while at the same time covering most of the locations in training for context map learning. In the original S-GAN work, different datasets were used for training and evaluation. To ensure a fair comparison, our approach uses the same data regime for our proposed method and the baselines. We provided semantic label annotations of the reference image for walkable areas and obstacles, i.e. $\mathcal{T} = \{\mathrm{walkable}, \mathrm{obstacle}\}$. They are visualized in Table I.

C. Implementation Details

For the loss weights during generator training of our models, we use $w_1 = 0.05$, $w_2 = 0.05$, $w_3 = 0.5$, $w_4 = 1$, $w_5 = 0.1$. We use a batch size of 32 and we train the model over 200 epochs for convergence. For the training process, we alternate updates to the discriminator and the generator per batch using an Adam optimizer for both with a learning rate of 5e-3. For map patches, a size of $H_{patch} = W_{patch} =$ 10px is used. This hyperparameter depends, in practice, on the dataset and resolution of the reference image in practice. For the $\mathrm{MapEncoder}_R$, we used the convolutional layers of a ResNet18 model and append a 3-layer convolutional head with $[10, 10, 2]$ output channels (and quadratic kernels of size $[1, 3, 3]$) respectively. For the semantic labels, prediction, and reconstruction encoders we have used nonlinear (ReLU) convolutional modules: feature count of $[5, 5]$ and kernel size of $[5 \times 5, 5 \times 5]$ for $\mathrm{MapDecoder}_S$, $[7, 5, 10]$ feature sizes and pixelwise convolutions for $\mathrm{MapDecoder}_P$, and a single 5-feature pixelwise operator for $\mathrm{MapDecoder}_R$.

Once training is done, we use the model that had the best performance on the validation set for reporting results on the test set. For all S-GAN models, including our variant based on latent maps, the evaluation is based on letting the generator draw one sample. All decoder networks have been implemented as CNNs. The full implementation will be made available upon acceptance of this work.

D. Results

Similarly to [5], we evaluate the performance of our model on trajectory prediction using two metrics: Average Displacement Error (ADE) and Final Displacement Error (FDE) which are also known as Mean L2 Error (ML2) and Final L2 error (FL2) [3]. The ADE, given as

$$\mathrm{ADE} = \frac{1}{N \cdot P} \sum_{i=1}^{N} \sum_{t=1}^{P} ||\mathbf{y}_{i,t} - \hat{\mathbf{y}}_{i,t}|| \ ,$$

averages the error between every position in the prediction and the ground truth. The FDE, given as

$$\mathrm{FDE} = \frac{1}{N} \sum_{i=1}^{N} ||\mathbf{y}_{i,P} - \hat{\mathbf{y}}_{i,P}|| \ ,$$

denotes the error at the last predicted position. Both metrics are averaged over the entire test set.

The results are visualized in Table II. Overall, use of the newly proposed context maps strongly improves the results compared to merely using different variants of S-GAN. This is mainly due to the fact that it becomes easier to tailor the prediction to the data and the environment. The qualitative differences to the original S-GAN results are mainly due to the fact that we learn directly on pixel space and have a different data preparation and augmentation process. Particularly the absence of a homography unifying the scale of the trajectories makes it more challenging to properly predict trajectories at different scales without location-specific memory.

Our work confirms that the contribution of the pooling module is minor (for the prediction task) compared to storing context. This, however, is partially due to the datasets not containing enough interactions such as near collisions. Furthermore, the strong improvements on the ETH sequence compared to not using context maps confirms our hypothesis that learned maps are particularly beneficial in situations with dataset imbalances and changing viewpoints.

E. Qualitative Examples

Several example trajectories from the ZARA2 and HOTEL sequences are visualized in Figure 5. These examples also involve some failure cases: In the ZARA2 sequence, the network may assume that pedestrians are likely to enter the store when close to the entrance as this is a frequently observed behaviour at that location. Overall, however, this ability of the proposed predictor to capture typical trajectories and behaviours at a given location results in a better prediction accuracy. At the same time, the auxiliary loss terms help to avoid over-fitting during training.

V. CONCLUCSION

In this work, we presented Deep Context Maps, a map learning approach for agent trajectory forecasting. We demonstrated how this approach can be integrated into a state-of-the-art predictor and that it achieves significant improvements on the agent trajectory forecasting task. Overall, maps promise to avoid over-fitting to the location of the training set in deep-learning based inference tasks for autonomous systems that are deployed in a big variety of diverse environments.

REFERENCES

[1] 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 *CVPR*, 2017, pp. 336–345.

[2] V. Kosaraju, A. Sadeghian, R. Martín-Martín, I. Reid, S. H. Rezatofighi, and S. Savarese, "Social-BiGAT: Multimodal Trajectory Forecasting using Bicycle-GAN and Graph Attention Networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019, pp. 137–146.

[3] A. Sadeghian, V. Kosaraju, A. Sadeghian, N. Hirose, H. Rezatofighi, and S. Savarese, "SoPhie: An Attentive GAN for Predicting Paths Compliant to Social and Physical Constraints," in *CVPR*, 2019, pp. 1349–1358.

[4] M.-F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan, *et al.*, "Argoverse: 3d tracking and forecasting with rich maps," in *CVPR*, 2019.

[5] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social GAN: Socially Acceptable Trajectories With Generative Adversarial Networks," in *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 2255–2264.

[6] D. Koller, K. Daniilidis, and H.-H. Nagel, "Model-based object tracking in monocular image sequences of road traffic scenes," *International Journal of Computer Vision*, vol. 10, no. 3, pp. 257–281, 1993.

[7] D. Helbing and P. Molnár, "Social force model for pedestrian dynamics," *Physical Review E*, vol. 51, no. 5, 1995.

[8] S. Lefèvre, D. Vasquez, and C. Laugier, "A survey on motion prediction and risk assessment for intelligent vehicles," *ROBOMECH journal*, vol. 1, no. 1, p. 1, 2014.

[9] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras, "Human Motion Trajectory Prediction: A Survey," *arXiv preprint:1905.06113*, 2019.

[10] D. Ridel, E. Rehder, M. Lauer, C. Stiller, and D. Wolf, "A Literature Review on the Prediction of Pedestrian Behavior in Urban Scenarios," in *Proceedings of the International Conference on Intelligent Transportation Systems (ITSC)*, 2018.

[11] A. Vemula, K. Muelling, and J. Oh, "Modeling cooperative navigation in dense human crowds," in *Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2017.

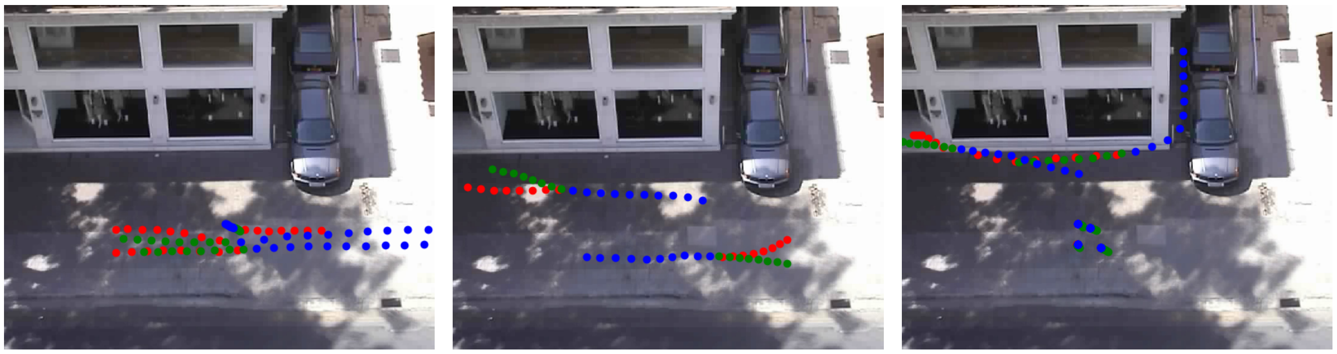
[12] N. Radwan, A. Valada, and W. Burgard, "Multimodal Interaction-aware Motion Prediction for Autonomous Street Crossing," *arXiv:1808.06887*, 2018.

[13] M. Pfeiffer, G. Paolo, H. Sommer, J. Nieto, R. Siegwart, and C. Cadena, "A Data-driven Model for Interaction-aware Pedestrian Motion Prediction in Object Cluttered Environments," in *Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2018, pp. 5921–5928.

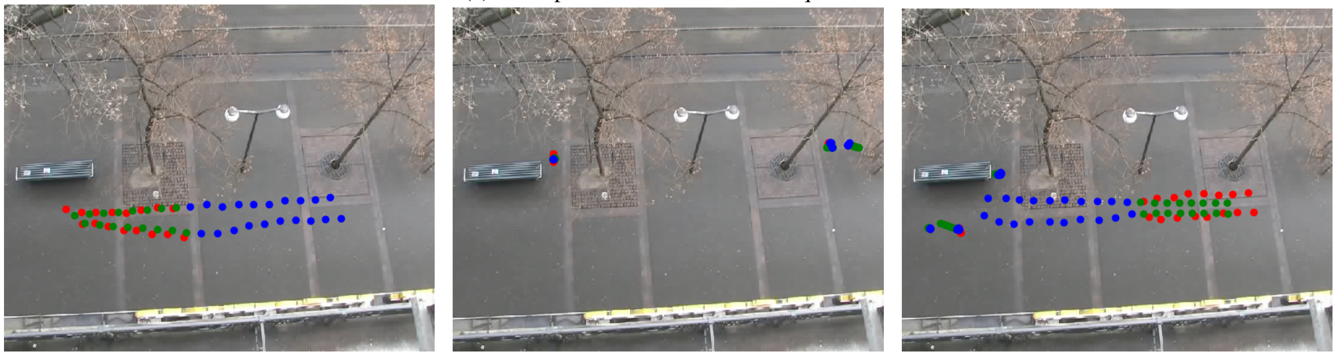
[14] B. Ivanovic and M. Pavone, "The trajectron: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs," in *ICCV*, 2019, pp. 2375–2384.

[15] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human Trajectory Prediction in Crowded Spaces," in *CVPR*, 2016, pp. 961–971.

[16] J. Wiest, M. Höffken, U. Kreßel, and K. Dietmayer, "Probabilistic trajectory prediction with Gaussian mixture models," in *Proceedings of the Intelligent Vehicles Symposium (IV)*, 2012, pp. 141–146.



(a) Examples from the ZARA2 sequence.



(b) Examples from the HOTEL sequence.

Fig. 5: **Example Predictions.** Several selected example trajectories (red) from different scenes with corresponding predictions by the network (green) and the observed past trajectories (blue) that were used for generating the predictions.

[17] P. Coscia, F. Castaldo, F. A. Palmieri, A. Alahi, S. Savarese, and L. Ballan, "Long-term path prediction in urban scenarios using circular distributions," *Image and Vision Computing*, pp. 81–91, 2018.

[18] D. Petrich, T. Dang, D. Kasper, G. Breuel, and C. Stiller, "Map-based long term motion prediction for vehicles in traffic environments," in *Proceedings of the International Conference on Intelligent Transportation Systems (ITSC)*, 2013, pp. 2166–2172.

[19] L. Ballan, F. Castaldo, A. Alahi, F. Palmieri, and S. Savarese, "Knowledge Transfer for Scene-Specific Motion Prediction," in *ECCV*, 2016.

[20] G. Habibi, N. Jaipuria, and J. P. How, "Context-Aware Pedestrian Motion Prediction In Urban Intersections," *arXiv preprint:1806.09453*, 2018.

[21] J. F. P. Kooij, F. Flohr, E. A. I. Pool, and D. M. Gavrila, "Context-Based Path Prediction for Targets with Switching Dynamics," *IJCV*, pp. 239–262, 2018.

[22] A. Sadeghian, F. Legros, M. Voisin, R. Vesel, A. Alahi, and S. Savarese, "CAR-Net: Clairvoyant Attentive Recurrent Network," in *ECCV*, 2018, pp. 162–180.

[23] H. Xue, D. Q. Huynh, and M. Reynolds, "SS-LSTM: A Hierarchical LSTM Model for Pedestrian Trajectory Prediction," in *Proceedings of the Winter Conference on Applications of Computer Vision (WACV)*, 2018, pp. 1186–1194.

[24] K. M. Kitani, B. D. Ziebart, J. A. Bagnell, and M. Hebert, "Activity Forecasting," in *ECCV*, 2012, pp. 201–214.

[25] D. Ridel, N. Deo, D. Wolf, and M. Trivedi, "Scene Compliant Trajectory Forecast With Agent-Centric Spatio-Temporal Grids," *Robotics and Automation Letters*, vol. 5, no. 2, pp. 2816–2823, 2020.

[26] H. O. Jacobs, O. K. Hughes, M. Johnson-Roberson, and R. Vasudevan, "Real-Time Certified Probabilistic Pedestrian Forecasting," *Robotics and Automation Letters*, vol. 2, no. 4, pp. 2064–2071, 2017.

[27] A. Rudenko, L. Palmieri, and K. O. Arras, "Joint Long-Term Prediction of Human Motion Using a Planning-Based Social Force Approach," in *Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2018, pp. 4571–4577.

[28] E. Rehder, F. Wirth, M. Lauer, and C. Stiller, "Pedestrian Prediction by Planning Using Deep Neural Networks," in *Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2018.

[29] W. Luo, B. Yang, and R. Urtasun, "Fast and Furious: Real Time End-to-End 3D Detection, Tracking and Motion Forecasting With a Single Convolutional Net," in *CVPR*, 2018, pp. 3569–3577.

[30] F. Ramos and L. Ott, "Hilbert maps: Scalable continuous occupancy mapping with stochastic gradient descent," *The International Journal of Robotics Research*, vol. 35, no. 14, pp. 1717–1730, 2016.

[31] S. Gupta, J. Davidson, S. Levine, R. Sukthankar, and J. Malik, "Cognitive Mapping and Planning for Visual Navigation," in *CVPR*, 2017, pp. 7272–7281.

[32] E. Parisotto and R. Salakhutdinov, "Neural Map: Structured Memory for Deep Reinforcement Learning," *arXiv preprint:1702.08360*, 2017.

[33] J. Zhang, L. Tai, J. Boedecker, W. Burgard, and M. Liu, "Neural SLAM: Learning to Explore with External Memory," *arXiv preprint:1706.09520*, 2017.

[34] G. Avraham, Y. Zuo, T. Dharmasiri, and T. Drummond, "EMPNet: Neural Localisation and Mapping Using Embedded Memory Points," in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2019, pp. 8119–8128.

[35] S. Yi, H. Li, and X. Wang, "Pedestrian Behavior Understanding and Prediction with Deep Neural Networks," in *ECCV*, 2016, pp. 263–279.

[36] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Nets," in *NeurIPS*, 2014, pp. 2672–2680.

[37] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[38] Y. Benezeth, P.-M. Jodoin, B. Emile, H. Laurent, and C. Rosenberger, "Review and evaluation of commonly-implemented background subtraction algorithms," in *ICPR*, 2008, pp. 1–4.

[39] M. Elad, *Sparse and redundant representations: from theory to applications in signal and image processing*. Springer, 2010.

[40] S. Pellegrini, A. Ess, K. Schindler, and L. van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2009, pp. 261–268.

[41] A. Lerner, Y. Chrysanthou, and D. Lischinski, "Crowds by Example," *Computer Graphics Forum*, vol. 26, pp. 655–664, 2007.