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

Igor Gilitschenski[1], Guy Rosman[2], Arjun Gupta[3], Sertac Karaman[3], 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 (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 maps can significantly improve predictor accuracy and be additionally boosted by providing even 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 operating in cluttered environments. 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, particularly 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 trajectory decision making process involves and depends on a high variety of factors including the surrounding environment and the interaction with it.

Early approaches for modelling pedestrian motion involve the use of constant-acceleration models [1]. Several physically inspired models, most prominently the social forces model [2], incorporate consideration of pedestrian group dynamics. The use of deep learning based approaches for pedestrian prediction tasks allowed for more complex models of the future trajectory enabling implicitly learning typical phenomena of social interaction [3], [4]. However, these models do not take take the environment into account beyond considering the motions of other agents.

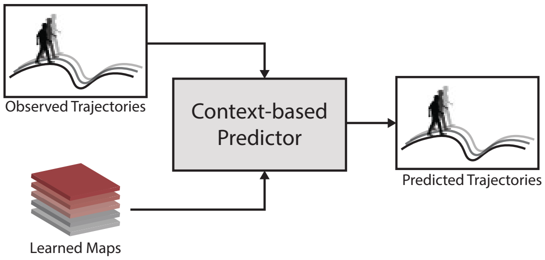


Fig. 1: **Deep Context Map.** The proposed new prediction model uses location specific learned maps to encode visual, temporal, and semantic scene context information. At test time, these maps are used to improve the prediction accuracy.

More recently, several deep-learning based approaches incorporated context to improve prediction accuracy [5], [6], [7]. This was usually achieved by using a top-down view of the scenery into the prediction network. This helps to improve prediction performance but fails to account for information that is not available through visual cues or make use of structure available through maps. For instance, the raw trajectory data contains a lot of information about local norms and the environment. Trajectories implicitly encode phenomena such as common paths, potential obstacles, and traffic flow and could be a rich source of non-visual information. Thus, it is an open problem how to incorporate different types of subtle information into a context representation for the prediction task.

In this paper, we address this gap by proposing an approach for learning a predictor together with context maps through weak-supervision from past agent trajectories. Instead of presuming a predetermined structure, we train the maps as latent entities that can not only explain visual 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. 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.

*This work has been supported by the Toyota Research Institute. It, however, reflects solely the opinions and conclusions of its authhors and not TRI or any other Toyota entity.

[1]MIT Computer Science and Artificial Intelligence Lab (CSAIL), igilitschenski@mit.edu, rus@csail.mit.edu

[2]Toyota Research Institute, guy.rosman@tri.global

[3]MIT Laboratory for Information and Decision Systems (LIDS), argupta@mit.edu, sertac@mit.edu

- 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

An extensive body of research considers the problem of behaviour prediction with an emphasis on traffic agents and crowds in public spaces. An early approach casting pedestrian prediction as inverse optimal control problem was presented in [8]. In [9], previously observed motion patterns are used to estimate a probability distribution as motion model. A mixed Markov-chain model is used in [10] and compared against pure Markov chain and hidden Markov chain based approaches. Observations of previous trajectories are used in [11] for estimating circular distributions at different locations which are combined into a smooth path prediction. In [12] context features from the environment (such as traffic light status and distance to curbside) are incorporated into a Gaussian Process based framework for pedestrian trajectory prediction. An existing map with annotated traffic lanes and centerlines is used for prediction in [13] based on a Kalman Filter framework for predicting vehicle motion. The work [14] 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 [15] future trajectories of all interacting agents are modeled by learning social interactions from real data using a Gaussian Process model. A Variational Gaussian Mixture Model is used in [16] to learn on-board pedestrian behaviour around vehicles. In contrast to these works, our approach makes use of deep learning to account for complexity of the trajectory forecasting process. Furthermore, we do not only allow for providing semantic cues at inference time but also can use them during the training process to inform our maps.

More recently, neural network approaches have been applied to the prediction task putting an emphasis on directly learning social interactions or considering environment semantics. In [17], a single convolutional network is used to combine detection, tracking, and motion prediction and [18] uses an interaction-aware trajectory prediction network (combined with traffic light recognition) to asses the safety of a street intersection for crossing. In [19] static obstacles and surrounding pedestrians are explicitly modeled for improving the forecasting task. Social Pooling modules are proposed in [3], [4], to allow for reasoning about other pedestrians trajectories. While these approaches focus on interaction awareness, we demonstrate that a better modelling of the environment has a stronger influence on the accuracy of the prediction than proper modelling of social interactions.

Several recent approaches actively consider context information for trajectory prediction. A deep learning-based inverse optimal control approach for predicting multiple interacting agents is presented in [5]. It uses scene context derived from image-based features and other sensory data. Similarly, visual context cues are used in [20], [6], [21], [22], [7] which usually take an image of the scenery as an additional input to the network. In contrast to these approaches, we use an image of the scenery to guide the map learning process. One motivation for our approach is the use in autonomous driving, where real-time top-view data showing other agents may not be available at inference time. A location specific bias tensor is used for pedestrian prediction in [23]. This approach uses a displacement volume as a network input and proposes a bias map of the size of that volume for improving prediction. Unlike our work, it considers prediction only on one scene and has only one bias value per location that is shared across channels. There is also no specific supervision for these biases in the training process making it incapable of encoding more complex scenery semantics or being applicable to a diverse set of scenarios.

Our work draws some inspiration from several approaches that encode map representations within a learning pipeline. An early approach of map learning was presented in [24], where a Gaussian Process is used for occupancy mapping without a priori discretization of the world into grid cells. An explicit destination network is used in [25] to model a grid of potential destinations for subsequent planning based trajectory prediction. In [26], a differentiable mapper is used to create a multiscale belief of the world in the agent's coordinate frame which is then directly used as input in an also differentiable hierarchical planner. Similarly, the deep active localization approach in [27] generates control actions from a map and sensor data in an end-to-end framework that involves learned perception and policy modules. We build upon these works by proposing the first implicit map building approach to improve the trajectory prediction task. Unlike these approaches, we use merely weak supervision to provide some structure of the maps and asses them in terms of their utility.

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 $\mathbf{X} = \{X_1, \dots, X_N\}$ and a learned location specific map M_l, i.e.

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

such that $\hat{\mathbf{Y}}$ approximates the ground truth trajectories, $\mathbf{Y}$, as closely as possible. The trajectories are represented as sequences $X_i = \{\mathbf{x}_{i,t} \in \mathbb{R}^2 \,|\, t = 1, \dots, O\}$ and $Y_i = \{\mathbf{y}_{i,t} \in \mathbb{R}^2 \,|\, t = 1, \dots, P\}$ with observation horizon O and prediction horizon P. The neural network representing f is

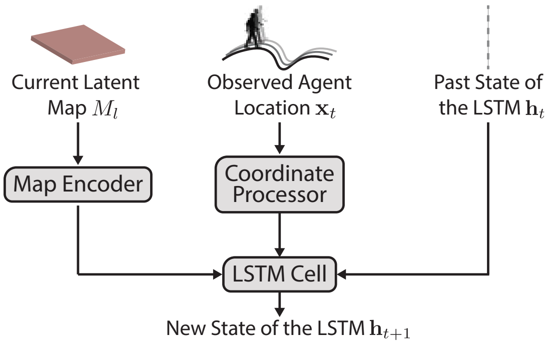

Current Latent Map M_l
Observed Agent Location $\mathbf{x}_t$
Past State of the LSTM $\mathbf{h}_t$
Map Encoder
Coordinate Processor
LSTM Cell
New State of the LSTM $\mathbf{h}_{t+1}$

Fig. 2: **LSTM Input.** The first LSTM cell in the generator network that is used for generating predictions sees a preprocessed variant of the map and the observed agent location. The preprocessing of the map happens using a CNN encoder. The trajectory is preprocessed using a MLP. The parameters of both preprocessor steps are learned.

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 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 (see Fig. 3).

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) [28] have been proposed demonstrating the capability for covering a variety of different plausible trajectories [4], [21], [6]. 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) [4]. However, 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) [29] network which is broadly used by several of the above-mentioned art 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:

$$\begin{aligned} \mathbf{e}_c &= \mathrm{MLP}(\mathbf{x}_{i,t}) \ , \\ \mathbf{h}_t &= \mathrm{LSTM}(\mathbf{h}_{t-1}, \, \mathbf{e}_c) \ , \end{aligned} \tag{1}$$

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 as visualized in Fig. 2. 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

$$\begin{aligned} \mathbf{e}_m &= \mathrm{CNN}(C_{i,t}) \ , \\ \mathbf{e}_c &= \mathrm{MLP}(\mathbf{x}_{i,t}) \ , \\ \mathbf{h}_t &= \mathrm{LSTM}(\mathbf{h_{t-1}}, \mathbf{e}_c, \mathbf{e_{context}}) \end{aligned} \tag{2}$$

Where CNN is the Convolutional Neural Network to encode the context map, and $C_{i,t}$ is a patch of the context map around the location $\mathbf{x}_{i,t}$. 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 [4, eq. (4)]).

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

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

B. Map Encoder

The maps are stored as a set of trainable weights in a map encoder module. During the network's forward pass, any locations $\mathbf{x}$ in the considered scene can be used to extract a patch of size $H_{patch} \times W_{patch}$ denoted as

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

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

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. We enforce this constraint by including a decoder network and provide an image explanation loss to reconstruct the reference image according to

$$\mathcal{L}_{\text{image}}(l) = ||I_l - \text{MapDecoder}_{\text{R}}(M_l)|| \ .$$

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. We formulate the loss as the difference between hand-annotated semantic labels and the decoded annotation

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

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.

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

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

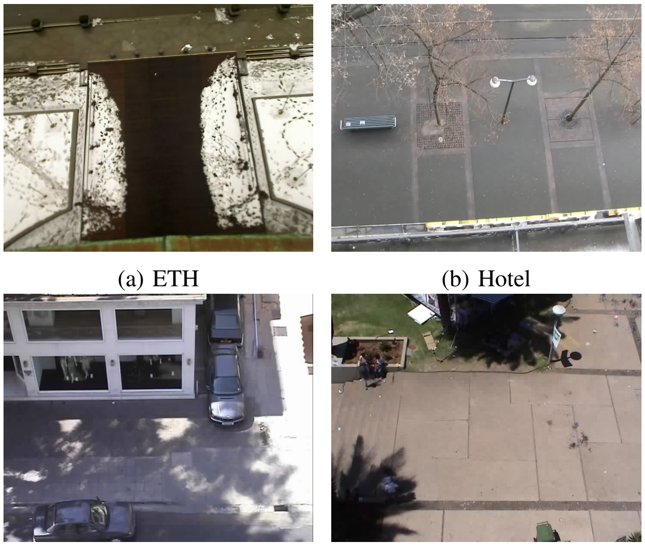
to the latent map. Then, we compute the norm by treating the resulting output as a vector.

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.

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. 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.

We then train the context map by taking a weighted average of the gradients computed in the three networks above. The goal is that the training process enforces the idea that the context map should be able to inform 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


(a) ETH

(b) Hotel

(c) Zara1

(d) Students3

Fig. 3: **Reference Images.** The reference images that are used as supervisory reconstruction labels for our networks are generated by applying a median filter to the video data in the datasets.

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

2) Discriminator Step: 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 discriminators 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).

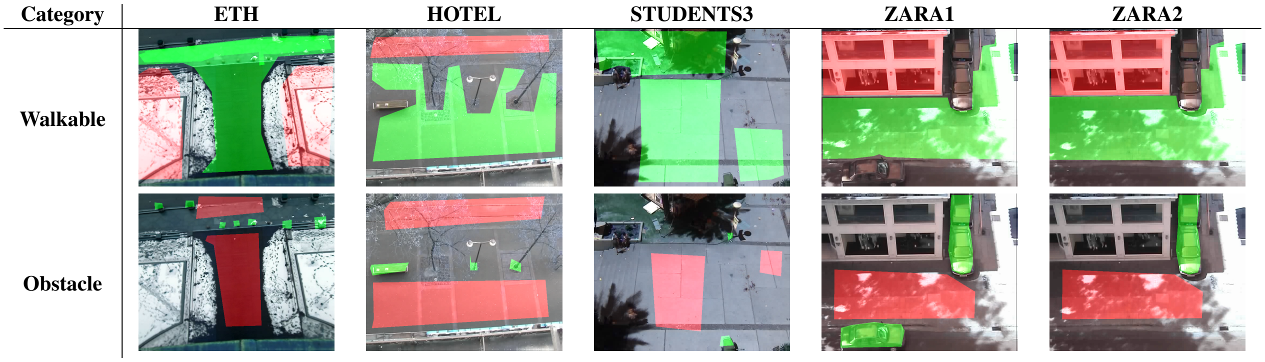


TABLE I: **Annotated labels.** The labels are given in terms of positive examples (green) and negative examples (red) encoded as 1 and -1 respectively. They do not cover the full corresponding semantic area in every reference image. We demonstrate that the information encoded in those labels can still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

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:

- **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.
- **S-GAN-P:** We compare our approach against the full S-GAN model including the 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.
- **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.
- **Ours:** Our full model involving all loss terms described above.
- **Ours no pooling:** Our full model without the pooling module in the generator to evaluate the relative contribution of pooling.
- **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 and a corresponding reference image of the scenery. Thus, it was evaluated on ETH and UCY standard benchmark datasets.

ETH Dataset [30] 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 [31]. 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. 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.

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

TABLE II: **Prediction Results** given in terms of ADE (left) and FDE (right). 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.

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. For map patches, a size of $H_{patch} = W_{patch} = 10\text{px}$ is used. Once training is done, we use the model that had the best performance on the validation set for reporting results on the test set. All decoder networks have been implemented as a CNNs. The full implementations will be made available upon publication of this work.

D. Results

Similarly to [4], 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) [7]. 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.

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] O. Masoud and N. Papanikolopoulos, "A novel method for tracking and counting pedestrians in real-time using a single camera," *Transactions on Vehicular Technology*, vol. 50, no. 5, 2001.

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

[3] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human Trajectory Prediction in Crowded Spaces," in *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[4] 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.

[5] 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 *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[6] 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.

[7] A. Sadeghian, V. Kosaraju, A. Sadeghian, N. Hirose, H. Rezatofighi, and S. Savarese, "SoPhie: An Attentive GAN for Predicting Paths Compliant to Social and Physical Constraints," in *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[8] B. D. Ziebart, N. Ratliff, G. Gallagher, C. Mertz, K. Peterson, J. A. Bagnell, M. Hebert, A. K. Dey, and S. Srinivasa, "Planning-based Prediction for Pedestrians," in *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, 2009.

[9] 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.

[10] A. Asahara, K. Maruyama, A. Sato, and K. Seto, "Pedestrian-movement Prediction Based on Mixed Markov-chain Model," in *Proceedings of the International Conference on Advances in Geographic Information Systems (SIGSPATIAL)*, 2011.

[11] 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*, 2018.

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

[13] 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.

[14] L. Ballan, F. Castaldo, A. Alahi, F. Palmieri, and S. Savarese, “Knowledge Transfer for Scene-Specific Motion Prediction,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[15] 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.

[16] N. Deo and M. M. Trivedi, “Learning and predicting on-road pedestrian behavior around vehicles,” in *Proceedings of the International Conference on Intelligent Transportation Systems (ITSC)*, 2017.

[17] 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 *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

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

[19] 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.

[20] J. F. P. Kooij, F. Flohr, E. A. I. Pool, and D. M. Gavrila, “Context-Based Path Prediction for Targets with Switching Dynamics,” *International Journal of Computer Vision*, 2018.

[21] A. Sadeghian, F. Legros, M. Voisin, R. Vesel, A. Alahi, and S. Savarese, “CAR-Net: Clairvoyant Attentive Recurrent Network,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[22] 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.

[23] S. Yi, H. Li, and X. Wang, “Pedestrian Behavior Understanding and Prediction with Deep Neural Networks,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[24] 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, 2016.

[25] 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.

[26] S. Gupta, J. Davidson, S. Levine, R. Sukthankar, and J. Malik, “Cognitive Mapping and Planning for Visual Navigation,” in *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[27] S. K. Gottipati, K. Seo, D. Bhatt, V. Mai, K. Murthy, and L. Paull, “Deep Active Localization,” *Robotics and Automation Letters*, vol. 4, no. 4, 2019.

[28] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative Adversarial Nets,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

[29] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Computation*, vol. 9, no. 8, 1997.

[30] 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.

[31] A. Lerner, Y. Chrysanthou, and D. Lischinski, “Crowds by Example,” *Computer Graphics Forum*, 2007.