Improving Pedestrian Prediction Models with Self-Supervised Continual Learning

Luzia Knoedler*, Chadi Salmi*, Hai Zhu, Bruno Brito and Javier Alonso-Mora

Abstract—**Autonomous mobile robots require accurate human motion predictions to safely and efficiently navigate among pedestrians, whose behavior may adapt to environmental changes. This paper introduces a self-supervised continual learning framework to improve data-driven pedestrian prediction models online across various scenarios continuously. In particular, we exploit online streams of pedestrian data, commonly available from the robot's detection and tracking pipeline, to refine the prediction model and its performance in unseen scenarios. To avoid the forgetting of previously learned concepts, a problem known as catastrophic forgetting, our framework includes a regularization loss to penalize changes of model parameters that are important for previous scenarios and retrains on a set of previous examples to retain past knowledge. Experimental results on real and simulation data show that our approach can improve prediction performance in unseen scenarios while retaining knowledge from seen scenarios when compared to naively training the prediction model online.**

Index Terms—**Continual Learning, Service Robotics, Trajectory Prediction, Human-Aware Motion Planning**
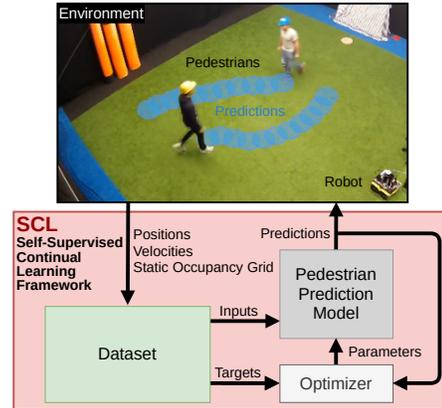

Fig. 1: Self-supervised Continual Learning (SCL) framework used to continuously improve data-driven pedestrian prediction models online across various scenarios.

I. INTRODUCTION

AUTONOMOUS mobile robots increasingly populate human environments, such as hospitals, airports and restaurants, to perform transportation, assistance and surveillance tasks [1]. In these continuously changing environments robots have to navigate in close proximity with pedestrians. To efficiently and safely navigate around them, robots must be able to reason about human behavior [2]. Predicting pedestrian trajectories is challenging, especially in crowded spaces where humans closely interact with their neighbors. This is the case, since the occurring interactions are complex, often subtle, and follow social conventions [3]. Furthermore, humans are influenced by the robot's presence [4], features of the static environment, such as its geometry or obstacle affordance, and various internal stimuli, such as urgency, which are difficult to measure [5], [6].

A large amount of research has been done on pedestrian prediction models [5]. Recently, the focus has mainly been on data-driven models which do not rely on hand-crafted functions and thus allow to capture more complex features and leverage large amounts of data. They address various aspects

The authors are with the Department of Cognitive Robotics, Delft University of Technology, Delft 2628 CD, The Netherlands. {l.knoedler; c.salmi; h.zhu; bruno.debrito; j.alonsomora}@tudelft.nl.
*Both authors contributed equally to this work.
Code: https://github.com/tud-amr/scl

of pedestrian behaviour such as stochasticity [7] and multimodality [8], [9]. Moreover, they consider the influence of static obstacles [2], interactions among pedestrians [3] and the robot's presence [10]. However, these models are trained offline using supervised learning and thus do not adapt to unseen behaviors or environments and may fail if the testing data distribution differs from the training data distribution.

These limitations can be overcome by continuously training pedestrian prediction models on new streams of data. Hurdles in applying supervised continuous learning to existing prediction models are the slow and expensive creation of labeled data sets or the lack of supervision [11]. Robots operating in the same environment as pedestrians can autonomously collect training examples based on the robot's never-ending stream of observations. If a robot can efficiently and autonomously collect examples, its internal prediction models can be updated on the fly and the robot can effectively adapt its behavior. However, neural networks are prone to forget previously learned concepts while sequentially learning new concepts [11]. This phenomenon is referred to as *catastrophic forgetting*. To overcome catastrophic forgetting, we use a *regularization* strategy, namely elastic weight consolidation (EWC) [12], to selectively slow down learning for important model parameters, in combination with *rehearsing* a small set of examples from previous tasks.

The main contribution of this work is therefore the introduction of a self-supervised continual learning framework that uses online streams of data of pedestrian trajectories to continuously refine data-driven pedestrian prediction models, see Fig. 1. Our approach overcomes catastrophic forgetting by

combining a regularization loss and a data rehearsal strategy.

We evaluate the proposed method in simulation, showing that our framework can improve prediction performance over baseline methods and avoid catastrophic forgetting, and in experiments with a mobile robot, showing that our framework can continuously improve a prediction model without the need for external supervision.

II. RELATED WORK

In this section, we describe relevant approaches for pedestrian motion prediction and continual learning.

A. Pedestrian Motion Prediction

There has been a vast amount of work devoted to pedestrian trajectory prediction [5]. Early works are mainly model-based, such as the well-known social force model (SFM) which uses attracting and repulsive potentials to model the social behaviours of pedestrians [13], and the velocity-based models which compute collision-free velocities for trajectory prediction [14], [15]. A limitation of these model-based approaches is that they only utilize handcrafted features, thus not being able to capture complex interactions in crowded scenarios. To overcome the limitation, recurrent neural networks (RNNs) have been used for human trajectory prediction, which allows to represent complex features and leverage large amounts of data [16]. Building on RNNs, [3] utilized LSTM networks to model time dependencies and employed a pooling layer to model interactions. [10] proposed a network model that is aware of the environment constraints. In addition, other network models have been developed to predict pedestrian trajectories, including Generative Adversarial Networks (GANs) [8], [17] and Conditional Variational Autoencoders (CVAEs) [18], [19]. Albeit being efficient, these models are usually trained and evaluated using (offline) bench-marking datasets [20], [21], [22], [23], which limits their online adaption to unseen scenarios. In this paper, we propose an approach to improve these models online by introducing a self-supervised continual learning framework.

B. Continual Learning

Continual learning (CL) addresses the training of a model from a continuous stream of data containing changing input domains or multiple tasks [24]. The goal of CL is to adapt the model continually over time while preventing new data from overwriting previously learned knowledge. Existing CL approaches that mitigate catastrophic forgetting for neural network-based models can be divided into three categories: architecture-, memory- and regularization-based [11], [25].

Architecture-based approaches change the architecture of the neural network by introducing new neurons or layers [26], [27], [28]. Intuitively, these approaches prevent forgetting by populating new untouched weights instead of overwriting existing ones. However, the model complexity grows with the number of tasks.

Memory-based approaches save samples of past tasks to rehearse previous concepts periodically [11]. There are two types of memory-based methods that differ in the way they memorize past experiences: *rehearsal* methods explicitly saving examples [29] and *pseudo-rehearsal* methods saving a generative model from which samples can be drawn [30]. The data stored in the memory of rehearsal methods can be randomly chosen or carefully selected [29], [31]. Some methods require task boundaries [29] while other methods can be applied to the task free setting [31]. Since memory-based approaches require a separate memory, they can become unsustainable with an increasing number of tasks.

Regularization-based approaches add a regularization term to the loss to prevent modification of model parameters. This can be done using basic regularization techniques, such as weight sparsification, early stopping, and dropout, or with more complex methods which selectively prevent changes in parameters that are important to previous tasks [11]. [12] introduced *Elastic Weight Consolidation* (EWC), a regularization approach limiting the plasticity of specific neurons based on their importance determined from the diagonal of the Fisher Information Matrix (FIM). To compute the FIM, clear task boundaries are required. Other regularization approaches focus on relaxing this assumption by automatically inferring task-boundaries [32], or by calculating the importance in an online fashion over the entire learning trajectory [33]. In contrast to other categories of approaches, these regularization-based methods do not require much computational and memory resources. However, one downside of regularization-based approaches is that an additional loss term is added, which can lead to a trade-off between knowledge consolidation and performance on novel tasks.

Most of the time, combining different continual learning strategies results in better performance [11]. Hence, in this paper, we employ the EWC regularization technique combined with a data rehearsal strategy to achieve continual learning to improve pedestrian prediction models.

III. PROBLEM FORMULATION

Throughout this paper, we denote vectors, $\mathbf{x}$, in bold lowercase letters, matrices, M, in uppercase letters, and sets, $\mathcal{X}$, in calligraphic uppercase letters.

We address the problem of continuously improving a trajectory prediction model online using streams of pedestrian data. This data includes the position and velocity of all n tracked pedestrians over time, and an occupancy map of the static environment $\mathcal{S}$. The position, velocity, and the surrounding static environment of the i-th pedestrian at time t are denoted by $\mathbf{p}_t^i = [p_{\mathrm{x},t}^i, p_{\mathrm{y},t}^i]$, $\mathbf{v}_t^i = [v_{\mathrm{x},t}^i, v_{\mathrm{y},t}^i]$, and $\mathcal{O}_t^{\mathrm{env},i} \subset \mathcal{S}$, respectively. The sub-scripts x and y indicate the x and y direction in the world frame. The super-script i denotes the *query-agent*, i.e., the pedestrian whose trajectory we want to predict.

Denote by $\mathcal{X}_t^i$ the observations acquired within a past time horizon t_{obs} for predicting pedestrian i's future trajectory, which typically includes its own states, the states of the other pedestrians and environment information. Further denote by $\hat{\mathcal{Y}}_t^i$ the predicted trajectory of pedestrian i over the future prediction horizon t_{pred}.

We seek a data-driven prediction model $\hat{\mathcal{Y}}_t^i = f_\theta(\mathcal{X}_t^i)$, with parameters θ, that best approximates the true trajectory $\mathcal{Y}_t^i$ across the entire previous stream of states for every tracked pedestrian $i \in \{1, \ldots, n\}$. The true trajectory $\mathcal{Y}_t^i$ will only become available in hindsight after observing the trajectory taken by pedestrian i during t_{pred}. Thus, we formulate the problem of continually learning a data-driven prediction model from past observations at time t as a regret minimization problem:

$$\min_\theta \sum_{i=1}^n \sum_{\tau=t-t_{\text{his}}}^t \mathscr{L}_{\text{pred}}(\hat{\mathcal{Y}}_\tau^i, \mathcal{Y}_\tau^i), \qquad (1)$$

where t_{his} is the entire elapsed time until t and $\mathscr{L}_{\text{pred}}(\hat{\mathcal{Y}}_\tau^i, \mathcal{Y}_\tau^i)$ is the regret at one past time step τ for pedestrian i, which will be described in later sections.

IV. METHOD

In this section, we introduce the Self-supervised Continual Learning (SCL) framework, an online learning framework to continually improve pedestrian prediction models. Section IV-A presents the overall structure of SCL, Section IV-B the prediction network architecture, Section IV-C the data aggregation and Section IV-D the model adaption.

A. Self-supervised Continual Learning

The SCL architecture consisting of two phases: a task aggregation and a model adaptation phase is presented in Fig. 2. Firstly, we use a prediction model which was pre-trained on publicly available datasets [20], [21] and aggregate new training examples using the surrounding pedestrians as experts (task aggregation) for a period of time T. Then, we update the prediction model using the aggregated data of the current task and a small constant sized coreset, which contains examples from previous tasks (model adaptation). During the model adaption phase, we apply a EWC loss to preserve the prediction performance on previous tasks. The two phases run alternately over time to create a continuous learning autonomous robot. During the task aggregation phase, we associate a new task to a new environment on which the model was previously not trained on. To distinguish between tasks, we will refer to the currently considered task as task_k. The previous tasks are referred to as $\text{task}_{0:k-1}$ where the subscript 0 refers to the initial task.

B. Prediction Network Architecture

To evaluate our online learning framework we use a data-driven pedestrian prediction model building on [2]. Please note that our approach does not depend on which network model we use. However, the memory requirements scale linearly with the number of tasks and model parameters. Figure 3 shows the network model which uses three streams of information. The first input is the query-agent's velocity over an observation time window t_{obs}, $\mathbf{v}_{t-t_{\text{obs}}:t}^i$, which enables the model to capture the pedestrian's dynamics. The second input is the occupancy grid information $O_{t-t_{\text{obs}}:t}^{\text{env,i}}$ that contains information about static obstacles centered on the query-agent. In contrast to [2], the third input is a vector containing information about the relative position and velocity of surrounding pedestrians $O_{t-t_{\text{obs}}:t}^{\text{social,i}}$. This adaption was done because the model using an angular pedestrian grid, presented in [2], has shown difficulties learning social interactions [19]. For one neighbor pedestrian j the vector including the relative measurements to the query-agent i at time t is

$$\mathbf{e}_t^{i,j} = [\mathbf{p}_t^j - \mathbf{p}_t^i, \mathbf{v}_t^j - \mathbf{v}_t^i].$$

Thus, the information vector at time t is defined as

$$O_t^{\text{social,i}} = [\mathbf{e}_t^{i,1}, \ldots, \mathbf{e}_t^{i,i-1}, \mathbf{e}_t^{i,i+1}, \ldots, \mathbf{e}_t^{i,n}].$$

Hence, the information used for trajectory prediction of pedestrian i is $\mathcal{X}_t^i = (\mathbf{v}_{t-t_{\text{obs}}:t}^i, O_{t-t_{\text{obs}}:t}^{\text{env,i}}, O_{t-t_{\text{obs}}:t}^{\text{social,i}})$, and the prediction model is given by $\hat{\mathbf{v}}_{t+1:t+t_{\text{pred}}}^i = f_\theta(\mathbf{v}_{t-t_{\text{obs}}:t}^i, O_{t-t_{\text{obs}}:t}^{\text{env,i}}, O_{t-t_{\text{obs}}:t}^{\text{social,i}})$, where the trajectory predictions are represented by a sequence of velocities, i.e. $\hat{\mathcal{Y}}_t^i = \hat{\mathbf{v}}_{t+1:t+t_{\text{pred}}}^i$. We use the permutation invariant $sort$ function as an attention mechanism by sorting the relative vectors of surrounding agents by euclidean distance [34]. To handle a variable number of pedestrians, only the closest n pedestrians are considered. For situations with fewer than n surrounding pedestrians, the relative vector of the closest pedestrian is repeated.

C. Task Aggregation

For each task k, SCL saves the inputs of the prediction model, $\mathcal{X}_t^i = (\mathbf{v}_{t-t_{\text{obs}}:t}^i, O_{t-t_{\text{obs}}:t}^{\text{env,i}}, O_{t-t_{\text{obs}}:t}^{\text{social,i}})$, in a buffer for each time step $t = \{-t_{\text{buff}}, \ldots, 0\}$ (see Fig. 2). Then, for each time step t, the ground truth velocity sequence $\mathbf{v}_{t+1:t+t_{\text{pred}}}^i$ is extracted in hindsight from the buffer and the corresponding input to the prediction model $\mathcal{X}_t^i$ (red arrows in Fig. 2). We aggregate the velocity vectors (Target) together with the corresponding model inputs (Input) and store them in the aggregated task dataset $\mathcal{D}_k$ as an example. The examples are aggregated as a sequence. As we use a recurrent prediction model and train the model with truncated back-propagation through time t_{tbptt}, we only aggregate sequences of examples with a length of $t_{\text{buff}} = t_{\text{pred}} + t_{\text{tbptt}}$. We collect training examples for T seconds.

D. Model Adaption

We present the overall SCL procedure in Algorithm 1. For each task, we aggregate a dataset $\mathcal{D}_k$ over T seconds. Then, the model is adapted using $\mathcal{D}_k$ and a set containing examples of previous tasks referred to as coreset $\mathcal{D}_{\text{coreset}}$. The *Coreset Rehearsal* strategy is applied to mitigate forgetting. Thus, the training dataset is defined as follows:

$$\hat{\mathcal{D}} = \mathcal{D}_k \bigcup \mathcal{D}_{\text{coreset}}.$$

In the model adaptation phase, SCL uses the training dataset $\hat{\mathcal{D}}$ to train the network for Q epochs. The training loss is composed by a prediction loss and a regularization loss to avoid catastrophic forgetting: $\mathscr{L}_{\text{train}} = \mathscr{L}_{\text{pred}} + \mathscr{L}_{\text{reg}}$. We

Fig. 2: Schematics of the SCL framework. The aggregation dataset is collected by extracting examples from the stream of tracked surrounding pedestrians (task aggregation). The prediction model is trained using the aggregated dataset and a separately saved coreset applying a EWC regularization to prevent catastrophic forgetting (model adaption).

Fig. 3: Pedestrian motion prediction model architecture.

define the prediction loss as the average norm between the predicted velocity sequence and the ground truth:

$$\mathscr{L}_{\text{pred}}(\hat{\mathcal{Y}}_t^i, \mathcal{Y}_t^i) = \frac{1}{t_{\text{pred}}} \sum_{\tau=t+1}^{t+t_{\text{pred}}} |\hat{\mathbf{v}}_\tau^i - \mathbf{v}_\tau^i|^2. \qquad (2)$$

We employ EWC [12] as regularization loss method to preserve prediction performance on the previous tasks (task$_{0:k-1}$) and overcome catastrophic forgetting. EWC penalizes the distance between the new model parameters, θ, and the previous task parameters, $\theta_{0:k-1}$, depending on their importance to keep the knowledge of previous tasks. After learning each task, EWC computes the corresponding importance parameter by using the diagonal elements of the FIM F, which are defined as:

$$F_{k,jj} = \frac{1}{|\mathcal{D}_k|} \sum_{\mathcal{X} \in \mathcal{D}_k} \left(\left. \frac{\delta \log f_\theta(\mathcal{X})}{\delta \theta_j} \right|_{\theta=\theta_k^*} \right)^2, \qquad (3)$$

where k represents the task number, $\mathcal{D}_k$ is the training data containing trajectories from task k, $f_\theta(\mathcal{X})$ is the predicted output of the network with parameters θ given data $\mathcal{X} \in \mathcal{D}_k$. The importance measure F_k is saved together with the net-

work weights θ_k. Based on $F_{0:k-1}$ and $\theta_{0:k-1}$ the following regularization term is added to the loss function:

$$\mathscr{L}_{\text{reg}}(\theta) = \sum_{l=0}^{k-1} \frac{\lambda}{2} F_l (\theta - \theta_l)^2, \qquad (4)$$

where θ is the current set of weights for the current task k and λ is the hyperparameter that dictates how important not forgetting the old task is compared to learning the new one. After the model adaptation phase is completed, we update the coreset with M examples of the latest task (task$_k$). Importantly the new examples replace existing ones to ensure the coreset remains of constant length N. We randomly select which examples to drop to update the coreset. After training the datasets $\mathcal{D}_k$ and $\hat{\mathcal{D}}$ are cleared.

V. RESULTS

In this section, we present quantitative and qualitative results in both simulation and real-world experiments.

A. Experimental Setup

The prediction model parameters are displayed in Fig. 3. We pre-train the prediction model on the ETH and UCY pedestrian datasets [20], [21] for 60 epochs. Our online learning framework will improve this pre-trained model based on the behavior of surrounding pedestrians. The applied hyperparameters are summarized in Table I. Note that although $t_{\text{obs}} = 0$, the past states are implicitly taken into account through the internal memory of the LSTMs. First, we evaluate our framework in simulation assuming full knowledge of the map and current states of all pedestrians. The pedestrian behaviour is simulated using the SFM [13] and Reciprocal Velocity Obstacle model (RVO) [35]. We train the prediction model incrementally on arbitrary orders of these environments.

Algorithm 1: The Self-supervised Continual Learning (SCL) framework

---

1 Load pretrained model: f_θ
2 Load map: $\mathcal{S}$
3 Initialize coreset: $\mathcal{D}_{\text{coreset}} \leftarrow \emptyset$
4 **for** $k = 0$ to ∞ **do**
5 $\quad$ Initialize the empty task dataset: $\mathcal{D}_k \leftarrow \emptyset$
6 $\quad$ Aggregate examples for T seconds as follows:
7 $\quad$ **for** $t = 0$ to T **do**
8 $\quad\quad$ Process pedestrian positions $\mathbf{p}_t^i$, velocities $\mathbf{v}_t^i$, and the occupancy grid $O_t^{\text{env,i}}$ to model inputs $\mathcal{X}_t^i$ and save them to a buffer for $i \in \{1, \ldots, n\}$
9 $\quad\quad$ Get examples from buffer: $\mathcal{E}_t = \{(\mathcal{X}_t^1, \mathcal{Y}_t^1), \ldots, (\mathcal{X}_t^n, \mathcal{Y}_t^n)\}$
10 $\quad\quad$ Update task dataset: $\mathcal{D}_k \leftarrow \mathcal{D}_k \bigcup \mathcal{E}_t$
11 $\quad$ **end**
12 $\quad$ Combine coreset and task: $\hat{\mathcal{D}} \leftarrow \mathcal{D}_k \bigcup \mathcal{D}_{coreset}$
13 $\quad$ Train prediction model f_θ on $\hat{\mathcal{D}}$ using EWC
14 $\quad$ Save EWC importances F_k and the updated parameters θ_k of task$_k$
15 $\quad$ Update coreset $\mathcal{D}_{coreset}$ with M random examples from $\mathcal{D}_k$
16 $\quad$ Clear $\mathcal{D}_k$, $\hat{\mathcal{D}}$ from memory
17 **end**

To evaluate how well our framework scales to complex scenarios with more pedestrians we rerun the above experiments in simulation with an increased number of pedestrians.
Then, we apply SCL in real-world experiments. Here, the true pedestrian behavior differs from the models assumed during simulation. To eliminate the perception-related errors as much as possible, we first test our framework with an optical tracking system (Optitrack) that provides pose information of all tracked pedestrians. We set up three scenarios to replicate the simulation environments. Finally, we evaluate our framework in an uncontrolled hall using only the on-board sensing and, a detection and tracking pipeline.

B. Baseline Methods

We evaluate our method against three baseline approaches in both simulation and real-world experiments:

1) **Offline:** The prediction model is trained offline on all tasks. This baseline represents a performance upper-bound assuming that all data is available.
2) **Vanilla:** The prediction model is trained using only the aggregated data and standard gradient descent without any regularization loss.
3) **EWC:** The prediction model is trained using only the aggregated data with EWC regularization, but without coreset rehearsal.

In simulation, we additionally consider the following baselines:

- **Coreset:** The prediction model is trained using the aggregated data and coreset data.
- **CV:** The human behaviour is predicted using the constant velocity model, no learning is applied.

The CV model was added since it was shown to outperform state-of-the-art data-based prediction models [36] and to

---

TABLE I: Hyperparameters.

time-step	0.2 s	# training epochs Q	250
task length T	200 s	learning rate	2×10^{-3}
buffer size t_{buff}	6 s	L2 regularization	5×10^{-4}
predict. time t_{pred}	3 s	EWC parameter λ	1×10^6
tbptt time t_{tbptt}	3 s	coreset size N/update size M	100/ 20
observ. time t_{obs}	0 s	validation set size L_v	100

enable robust navigation around humans [37]. A limitation of the CV model is that it does not consider obstacles.

Since our focus is on applying continual learning strategies to improve pedestrian prediction models on the fly without forgetting, we only change the learning strategy across baselines and keep the prediction network architecture fixed. Similar to other works on pedestrian prediction models, we use the average displacement (ADE) and final displacement error (FDE) as performance metrics [19], [34].
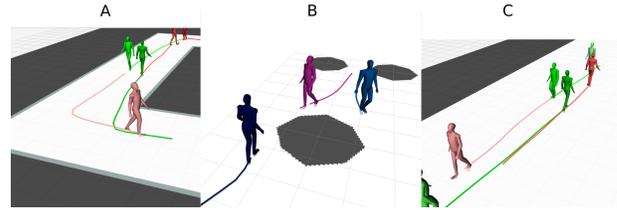
C. Tasks

Fig. 4: The considered simulation environments consist of (A) Square, (B) Obstacle, and (C) Hall environments.

We consider three distinct environments, i.e., tasks, displayed in Fig. 4:

1) **Square:** An infinite corridor setting with three pedestrians walking clockwise and three anticlockwise.
2) **Obstacles:** Pedestrians walking towards each other in an obstacle filled space.
3) **Hall:** Pedestrians walking towards each other in a hall while behaving cooperatively.

The scenarios were selected since they include encounters typically experienced in everyday situations. The specific environments were chosen to investigate social interactions (Hall), obstacle interactions (Obstacle) and semantic knowledge of the map (Square). To additionally evaluate our framework in scenarios with more interacting agents we consider the above environments with 10 and 20 pedestrians. For the obstacle-free environments, we use the open-source pedsim simulation framework[1] employing the SFM [13] to simulate the pedestrian behavior. For environments with static obstacles, we employ the RVO method [35][2] as pedestrians following the SFM may still collide with obstacles.

D. Simulation Results

We evaluate the prediction performance of the network model trained with our method (SCL) versus the baselines on
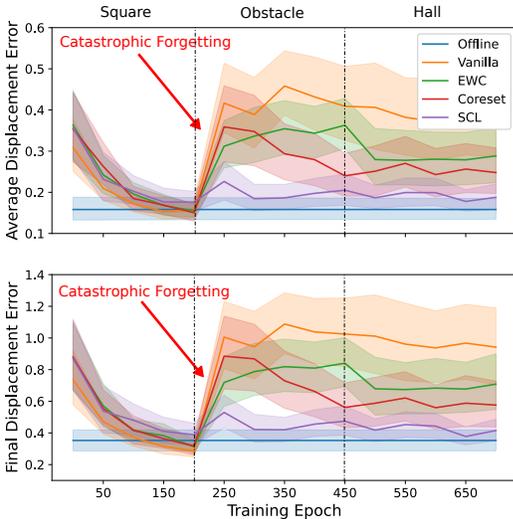
---

[1]https://github.com/srl-freiburg/pedsim_ros
[2]https://github.com/sybrenstuvel/Python-RVO2

Fig. 5: Prediction performance of models trained on square → obstacle → hall sequence evaluated on **only** the square scenario. Shows ADE (top) and FDE (bottom) of all training methods on the validation set of the square task, while learning new tasks. Note that the offline model is added for comparison purposes only.

different sequences of environments (square, obstacle, hall) starting from a pre-trained model. Each environment is observed for T seconds to create the aggregated dataset on which the model is trained. Thus, each environment corresponds to a new task. To compute the ADE/FDE performance metrics, we collect a validation set for each environment including L_v examples not used during training.

Table II reports the mean and standard deviation (std) of ADE/FDE evaluated at the end of the sequence on all three environments, under the columns denoted by *seq. end*. The columns denoted by *forgotten* report the mean and std of ADE/FDE increase of the prediction model on previous environments after training on new environments. It can be seen that SCL outperforms Vanilla. The significant increase in mean forgotten ADE/FDE for Vanilla indicates that naive online training over changing environments using standard gradient descent results in catastrophic forgetting. Our method is independent of sequence order, arriving at within ±0.02 of the same mean ADE/FDE for all orders.

To gain insight into where catastrophic forgetting occurs, we save the models trained for the sequence (square → obstacle → hall) after each training step and apply them to the validation set of the square scenario only. Figure 5 compares the performance of the different training methods on the square scenario validation set at each training step. By evaluating a single environment over time, we can clearly visualize when and how much the models degraded in prediction performance in the respective environment. For ease of comparison, the offline trained model is also plotted as a constant line. In the first section, all models are trained on the aggregated dataset of the square scenario and, as expected, the error measures decrease for all online learning methods reaching better perfor-

mance than offline trained prediction model due to overfitting. However, when changing from the square environment to the obstacle environment, the ADE/FDE performance quickly and drastically degrades for the Vanilla baseline (red arrow). It can be seen that using EWC to selectively slow down learning on important parameters helps to significantly mitigate the magnitude of the loss in ADE/FDE. Nevertheless, after two subsequent tasks, the EWC baseline performed ∼ 30% worse on FDE and ∼ 20% worse on ADE. Rehearsing a set of past examples enables to retain more knowledge after two subsequent tasks than applying EWC. Combining EWC and the coreset rehearsal as done in SCL helps to further mitigate forgetting. SCL was able to train in two subsequent scenarios while retaining knowledge of the initially experienced scenario.

We have performed pair-wise Mann-Whitney U tests between our proposed method and each baseline to evaluate the statistical significance of the presented results. Table III shows the p-values comparing the performance results (i.e., ADE and FDE) on each scenario for the obstacle → hall → square sequence. SCL significantly outperforms CV on all environments, the Vanilla baseline on all past environments, and EWC on one environment. Rehearsing alone achieves marginally worse results than SCL. Please note that the presented results consider a limited set of environments with limited complexity. We expect that as the number of scenarios and complexity increases, differences in performance between the baselines become significant.

E. Dense Scenarios

To evaluate how well our framework scales to complex scenarios with more pedestrians we employ the above simulation environments with increased numbers of pedestrians ($n = \{10, 20\}$). The results are presented in Table II. It can be seen that SCL scales well to dense scenarios with more agents achieving similar performance for 10 and 20 pedestrians. The forgotten ADE/FDE even decreases for some sequences, indicating that observing more pedestrians can improve the preservation of past experiences.

F. Real-world Results

We first evaluate our method in real-world experiments assuming perfect perception capabilities by using an external high-precision Optitrack tracking system. Secondly, we use the robot's on-board sensing capabilities combined with a detection and tracking pipeline.

1) Perfect Perception: To evaluate our framework using the Optitrack system, we set up three environments to replicate the ones considered in simulation (i.e., square, obstacle, cooperative). Each environment is observed for T seconds. Table IV reports quantitative results on two different sequence orders similar to Table II. Our framework significantly outperformed the Vanilla baseline on both metrics indicating that we can not only learn a prediction model from real human motion but also that we need to consolidate the learned knowledge. SCL was able to improve prediction performance and learn

TABLE II: Quantitative results of the CV prediction and Vanilla, EWC, Coreset and SCL training approaches for four environment sequences. The results for the dense scenarios are included as SCL-10 and SCL-20. The table lists the mean±standard deviation (std) of ADE / FDE for all environments at the sequence end under *seq. end* and the mean±std of the *forgotten* ADE / FDE, which refers to the average increase in ADE / FDE on previous environments across the learning sequence. All error measures are presented in meters.

Method	square → obstacle → hall		obstacle → square → hall		hall → obstacle → square		obstacle → hall → square	
	forgotten (mean±std)	seq. end (mean±std)	forgotten (mean±std)	seq. end (mean±std)	forgotten (mean±std)	seq. end (mean±std)	forgotten (mean±std)	seq. end (mean±std)
CV	+0.00±0.00/ +0.00±0.00	1.12±1.18/ 1.09±1.69	+0.00±0.00/ +0.00±0.00	1.25±1.29/ 1.10±1.65	+0.00±0.00/ +0.00±0.00	1.12±1.17/ 1.16±1.91	+0.00±0.00/ +0.00±0.00	1.26±1.31/ 1.24±1.92
Vanilla	+0.12±0.29/ +0.31±0.74	0.21±0.31/ 0.49±0.79	+0.10±0.23/ +0.27±0.62	0.21±0.27/ 0.47±0.63	+0.20±0.27/ +0.51±0.62	0.27±0.26/ 0.62 ±0.58	+0.18±0.20/ +0.52±0.51	0.26±0.21/ 0.63±0.51
EWC	+0.10±0.25/ +0.28±0.67	0.19±0.25/ 0.46±0.66	+0.05±0.13/ +0.12±0.37	0.17±0.13/ 0.37±0.35	+0.12±0.17/ +0.33±0.44	0.22±0.16/ 0.51±0.41	+0.10±0.18/ +0.27±0.47	0.21±0.19/ 0.48±0.45
Coreset	+0.03±0.09/ +0.09±0.27	**0.16±0.12/ 0.36±0.34**	+0.05±0.11/ +0.12±0.29	0.17±0.14/ 0.38±0.34	+0.03±0.10/ +0.08±0.25	0.17±0.13/ **0.36±0.28**	**+0.04±0.09/** +0.12±0.24	0.19±0.15/ 0.40±0.31
SCL	**+0.02±0.10/ +0.07±0.29**	0.16±0.14/ 0.36±0.40	**+0.01±0.08/ +0.04±0.20**	**0.15±0.12/ 0.34±0.27**	**+0.03±0.08/ +0.07±0.20**	**0.17±0.12/** 0.37±0.30	**+0.04±0.09/ +0.08±0.21**	**0.17±0.13/ 0.36±0.28**
SCL-10	+0.00±0.10/ +0.00±0.23	0.20±0.17/ 0.45±0.40	+0.01±0.12/ +0.03±0.25	0.20±0.16/ 0.44±0.37	+0.00±0.08/ +0.01±0.19	0.20±0.15/ 0.45±0.33	+0.01±0.10/ +0.01±0.26	0.20±0.14/ 0.44±0.33
SCL-20	+0.04±0.12/ +0.10±0.31	0.22±0.19/ 0.49±0.42	+0.02±0.10/ +0.05±0.25	0.20±0.16/ 0.46±0.37	+0.04±0.11/ +0.08±0.26	0.21±0.18/ 0.47±0.40	+0.03±0.12/ +0.06±0.28	0.22±0.18/ 0.50±0.41

TABLE III: Statistical Significance Analysis using the Mann-Whitney U test. Comparison of SCL's performance (i.e., ADE and FDE) against all baselines on each environment for the obstacle → hall → square sequence. Significant results are displayed in bold considering a 5% confidence-level.

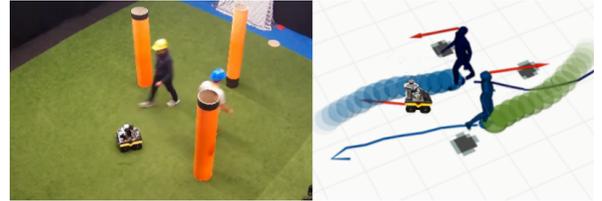| Method	obstacle		hall		square	
	ADE	FDE	ADE	FDE	ADE	FDE
CV	**p = 0.00**	**p = 0.00**	**p = 0.00**	**p = 0.00**	**p = 0.00**	**p = 0.00**
Vanilla	**p = 0.00**	**p = 0.00**	**p = 0.00**	**p = 0.00**	p = 0.97	p = 0.53
EWC	p = 0.06	**p = 0.02**	**p = 0.04**	**p = 0.01**	p = 0.65	p = 0.71
Coreset	p = 0.29	p = 0.09	p = 0.63	p = 0.68	p = 0.22	p = 0.33

Fig. 6: Real-world validation using an Optitrack system that streams the pedestrian states. The predicted pedestrian trajectories are depicted as green and blue disks.

certain concepts, such as avoiding crashing into walls, pillars, or fences. Figure 6 shows a qualitative example of the experiment, where our framework learns to avoid both static obstacles and pedestrians.

2) On-board Perception: We now evaluate our framework in an uncontrolled hall environment using the robot's detection and tracking pipeline (i.e., LiDAR and cameras). In Fig. 7 we show qualitative results of the experiments with a moving robot. The fact that the robot is constantly moving reduced the average collected trajectory length of the interacting pedestrians making the prediction problem harder. Thus, employing SCL in more dense environments is expected to further improve the resulting prediction performance. Nevertheless, the prediction model learned online when pedestrians are likely to take corners, by observing how real pedestrians walk in the environment. Note that the ETH and UCY datasets, on which our model was pre-trained, contain almost no interactions with static obstacles, yet our framework autonomously learns obstacle interactions. Furthermore, the occupancy map shown in Fig. 7 is generated by the robot itself using the depth information from its LiDAR. Thus, our framework can continuously learn in new and unseen environments autonomously.
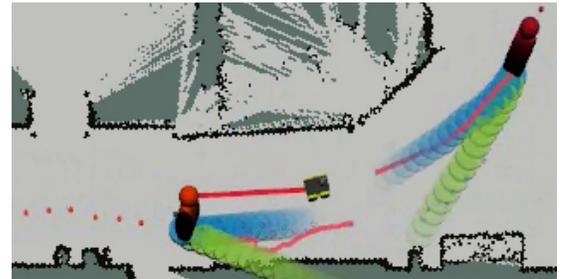

Fig. 7: Map view of the real-world application of SCL on moving robot using on-board perception. The green and blue disks depict the predicted trajectories employing the pre-trained model and the SCL-trained model, respectively. The red dotted lines depict the pedestrians' past trajectories. The pedestrians' and robot's future trajectories are shown as solid red lines.

VI. CONCLUSIONS & FUTURE WORK

This paper introduces a Self-supervised Continual Learning framework (SCL) to improve pedestrian prediction models using online streams of data. We combined Elastic Weight Consolidation (EWC) and the rehearsal of a small constant sized set of examples to overcome catastrophic forgetting. We showed through experiments that SCL significantly outperforms vanilla gradient descent and performs similarly to

TABLE IV: Quantitative results of Vanilla, EWC and SCL on real-world data collected using an optical tracking system. The table lists the mean±standard deviation (std) of ADE / FDE on all environments at the sequence end under *seq. end* and the mean±std of the *forgotten* ADE / FDE, which refers to the average increase in ADE / FDE on previous environments across the learning sequence. All error measures are presented in meters.

Method	square → obstacle → coop.		obstacle → square → coop.	
	forgotten (mean±std)	seq. end (mean±std)	forgotten (mean±std)	seq. end (mean±std)
Vanilla	+0.24±0.28/ +0.58±0.67	0.46±0.29/ 0.97±0.66	+0.21±0.26/ +0.50±0.64	0.45±0.29/ 0.94±0.63
EWC	+0.19±0.29/ +0.42±0.67	0.43±0.27/ 0.86±0.61	+0.12±0.23/ +0.31±0.58	0.41±0.25/ 0.87±0.56
SCL	**+0.04±0.21/ +0.13±0.50**	**0.36±0.23/ 0.73±0.56**	**+0.05±0.22/ +0.11±0.50**	**0.40±0.28/ 0.80±0.60**

offline trained models with full access to pedestrian data in all considered environments. Additionally, we showed in real-world experiments that our pedestrian prediction model can learn to generalize to new and unseen environments over time. Future work can investigate different methods to determine when the model should be updated, how different pedestrian behaviour types could be integrated into our framework and the integration of our approach with a motion planner to improve the interaction-awareness between pedestrians and the robot.

REFERENCES

[1] W. He, Z. Li, and C. P. Chen, "A survey of human-centered intelligent robots: issues and challenges," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 4, pp. 602–609, 2017.

[2] M. Pfeiffer, G. Paolo *et al.*, "A data-driven model for interaction-aware pedestrian motion prediction in object cluttered environments," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 5921–5928.

[3] A. Alahi, K. Goel *et al.*, "Social lstm: Human trajectory prediction in crowded spaces," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 961–971.

[4] P. Trautman and A. Krause, "Unfreezing the robot: Navigation in dense, interacting crowds," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 797–803.

[5] A. Rudenko, L. Palmieri *et al.*, "Human motion trajectory prediction: A survey," *The International Journal of Robotics Research*, vol. 39, no. 8, pp. 895–935, 2020.

[6] R. Löhner, "On the modeling of pedestrian motion," *Applied Mathematical Modelling*, vol. 34, no. 2, pp. 366–382, 2010.

[7] B. Ivanovic and M. Pavone, "The trajectron: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 2375–2384.

[8] J. Amirian, J.-B. Hayet, and J. Pettré, "Social ways: Learning multi-modal distributions of pedestrian trajectories with gans," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[9] B. Brito, B. Floor *et al.*, "Model predictive contouring control for collision avoidance in unstructured dynamic environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4459–4466, 2019.

[10] M. Pfeiffer, U. Schwesinger *et al.*, "Predicting actions to act predictably: Cooperative partial motion planning with maximum entropy models," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 2096–2101.

[11] T. Lesort, V. Lomonaco *et al.*, "Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges," *Information Fusion*, vol. 58, pp. 52–68, 2020.

[12] J. Kirkpatrick, R. Pascanu *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[13] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Physical review E*, vol. 51, no. 5, p. 4282, 1995.

[14] S. Kim, S. J. Guy *et al.*, "Brvo: Predicting pedestrian trajectories using velocity-space reasoning," *The International Journal of Robotics Research*, vol. 34, no. 2, pp. 201–217, 2015.

[15] A. Bera, S. Kim *et al.*, "Glmp-realtime pedestrian path prediction using global and local movement patterns," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016.

[16] S. Becker, R. Hug *et al.*, "Red: A simple but effective baseline predictor for the trajnet benchmark," in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018, pp. 0–0.

[17] A. Gupta, J. Johnson *et al.*, "Social gan: Socially acceptable trajectories with generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2255–2264.

[18] N. Lee, W. Choi *et al.*, "Desire: Distant future prediction in dynamic scenes with interacting agents," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 336–345.

[19] B. Brito, H. Zhu *et al.*, "Social-vrnn: One-shot multi-modal trajectory prediction for interacting pedestrians," in *2020 Conference on Robot Learning (CoRL)*, 2020.

[20] S. Pellegrini, A. Ess *et al.*, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *2009 IEEE 12th International Conference on Computer Vision*. IEEE, 2009, pp. 261–268.

[21] A. Lerner, Y. Chrysanthou, and D. Lischinski, "Crowds by example," in *Computer graphics forum*, vol. 26, no. 3. Wiley Online Library, 2007, pp. 655–664.

[22] H. Caesar, V. Bankiti *et al.*, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 621–11 631.

[23] M.-F. Chang, J. Lambert *et al.*, "Argoverse: 3d tracking and forecasting with rich maps," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8748–8757.

[24] M. Delange, R. Aljundi *et al.*, "A continual learning survey: Defying forgetting in classification tasks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[25] G. I. Parisi, R. Kemker *et al.*, "Continual lifelong learning with neural networks: A review," *Neural Networks*, vol. 113, pp. 54–71, 2019.

[26] J. Yoon, E. Yang *et al.*, "Lifelong learning with dynamically expandable networks," in *International Conference on Learning Representations*, 2018.

[27] C.-Y. Hung, C.-H. Tu *et al.*, "Compacting, picking and growing for unforgetting continual learning," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[28] X. Li, Y. Zhou *et al.*, "Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting," in *International Conference on Machine Learning*. PMLR, 2019, pp. 3925–3934.

[29] S.-A. Rebuffi, A. Kolesnikov *et al.*, "icarl: Incremental classifier and representation learning," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 2001–2010.

[30] H. Shin, J. K. Lee *et al.*, "Continual learning with deep generative replay," in *Advances in neural information processing systems*, 2017, pp. 2990–2999.

[31] R. Aljundi, M. Lin *et al.*, "Gradient based sample selection for online continual learning," *Advances in neural information processing systems*, vol. 32, 2019.

[32] R. Aljundi, K. Kelchtermans, and T. Tuytelaars, "Task-free continual learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 254–11 263.

[33] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence," in *International Conference on Machine Learning*. PMLR, 2017, pp. 3987–3995.

[34] A. Sadeghian, V. Kosaraju *et al.*, "Sophie: An attentive gan for predicting paths compliant to social and physical constraints," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1349–1358.

[35] J. Van den Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *2008 IEEE International Conference on Robotics and Automation*. IEEE, 2008, pp. 1928–1935.

[36] C. Schöller, V. Aravantinos *et al.*, "What the constant velocity model can teach us about pedestrian motion prediction," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1696–1703, 2020.

[37] C. I. Mavrogiannis, W. B. Thomason, and R. A. Knepper, "Social momentum: A framework for legible navigation in dynamic multi-agent environments," in *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, 2018, pp. 361–369.