Lifelong Vehicle Trajectory Prediction Framework Based on Generative Replay

Peng Bao, Zonghai Chen, Senior Member, IEEE, Jikai Wang, Member, IEEE, Deyun Dai, and Hao Zhao

Abstract—Accurate trajectory prediction of vehicles is essential for reliable autonomous driving. To maintain consistent performance as a vehicle driving around different cities, it is crucial to adapt to changing traffic circumstances and achieve lifelong trajectory prediction model. To realize it, catastrophic forgetting is a main problem to be addressed. In this paper, a divergence measurement method based on conditional Kullback-Leibler divergence is proposed first to evaluate spatiotemporal dependency difference among varied driving circumstances. Then based on generative replay, a novel lifelong vehicle trajectory prediction framework is developed. The framework consists of a conditional generation model and a vehicle trajectory prediction model. The conditional generation model is a generative adversarial network conditioned on position configuration of vehicles. After learning and merging trajectory distribution of vehicles across different cities, the generation model replays trajectories with prior samplings as inputs, which alleviates catastrophic forgetting. The vehicle trajectory prediction model is trained by the replayed trajectories and achieves consistent prediction performance on visited cities. A lifelong experiment setup is established on four open datasets including five tasks. Spatiotemporal dependency divergence is calculated for different tasks. Even though these divergence, the proposed framework exhibits lifelong learning ability and achieves consistent performance on all tasks.

Index Terms—Gaussian mixture model, conditional KL divergence, conditional GAN, generative replay, lifelong trajectory prediction.

I. INTRODUCTION

In autonomous driving, an ability to predict surrounding vehicles’ future trajectories accurately is a key to make appropriate decision. Therefore, many research works contribute on vehicle kinematics and interactive modelling [1]. Recent works also adopt data-driven approaches [2], [3], [4], [5], [6], [7], [8]. Benefitted from these ingenious works, precision of vehicle trajectory prediction on various open datasets has been promoted significantly.

However, in real application, an intelligent vehicle equipped with an autonomous driving system is supposed to visit varied road sections, cities or even countries. To guide the vehicle safely, the system is required to adapt to heterogeneous distribution of surrounding vehicles’ motion and interaction pattern and predict their future trajectories accurately. For this purpose, the system needs to learn new knowledge about emerging traffic environments continuously without forgetting old ones. In addition, with limited storage resource, the system cannot afford to store large amount of trajectory data. A candidate solution is to train different models for varied circumstances and update vehicle model configuration through real-time communication with cloud servers. This arises several problems. Firstly, privacy concern. Driving trajectories might contains personal intimate information and drivers would not want to upload it to the server. Secondly, connection stability. Connection to a server is not always accessible or stable enough to support large data transportation. Lastly, system independency and security. Frequent server communication will increase system burden and make it vulnerable to potential information intrusion. Keep it running independently would be more user-friendly. Toward full autonomous driving, the goal is to design a system that can drive at any circumstances. Driving performance is affected by many factors that formalize tremendous circumstances, which requires significant number of parameter configurations and model structures if trained specifically. Establishing a unified model, learning and promoting its performance continuously would be a more reasonable and promising approach. Towards this goal, continuous learning with limited storage resource to achieve good performance in all processed tasks is also called lifelong learning, where task means vehicle trajectory prediction under varying circumstances in our context and is unrelated with the life of vehicles. With limited memory, lifelong learning struggles to learn new knowledge contained in emerging tasks and update model configurations to achieve good performance on the new tasks. Meanwhile, considerable performance is remained on old tasks. In this way, the model is promoted to be more general to accommodate to more tasks, which in this context means a lifelong trajectory prediction model learns on-line and works on more and more circumstances. Unfortunately, most existing vehicle trajectory prediction models are trained and tested specifically for each dataset, which fails to accommodate to other different datasets if applied directly.

A crucial problem impeding lifelong learning is catastrophic forgetting. In different traffic circumstances, interaction pattern among vehicles varies, which can be interpreted as spatiotemporal dependency divergence. The divergence makes
a prediction model trained in old circumstances performs poorly in a new different circumstance. If trained in the new circumstances, the model would fit spatiotemporal dependency in the new one while forget that in the old circumstances, which makes it perform poorly in the old circumstances in turn. In summary, to perform consistently in all circumstances, the prediction model is required to learn new knowledges while not forgetting old ones.

Therefore, an ability of automatic divergence detection can also be a requirement for lifelong learning [9], [10], which also benefits selective knowledge learning. To measure spatiotemporal dependency divergence that arises from varied traffic circumstances, Mixture Density Networks (MDNs) are introduced to estimate conditional probability density function (CPDF) and conditional Kullback-Leibler divergence (CKLD) is computed through Monte-Carlo (MC) sampling. Then to realize lifelong trajectory prediction, a new framework based on conditional generative replay is proposed. The framework consists of two models, a generative memory and a task solver. The generative memory is designed to address catastrophic forgetting, a key challenge in lifelong learning. Through adversarial learning and distributions merging across different traffic circumstances, the generative memory replays trajectories conditioned on spatial configuration of vehicles. The replayed trajectories are used as an input to train a trajectory prediction model. As the generative memory is capable of generating trajectories within the same distribution of all recorded data, the prediction model achieves consistent performance on all processed tasks.

In summary, our contributions include:
- An innovative spatiotemporal dependency divergence measurement method is developed for two trajectory datasets.
- A novel generation model is first proposed to generate vehicle trajectories under heterogeneous circumstances.
- Based on generative replay, an initiate lifelong learning framework is introduced for vehicle trajectory prediction. Experimental results on five tasks have demonstrated that the proposed framework can achieve lifelong learning and realize satisfactory prediction performance on all processed tasks.

II. RELATED WORK

Vehicle trajectory prediction is a longstanding research topic and is becoming more important as the development of autonomous driving. Kinematics models [11], [12] are established to predict surrounding vehicles’ future trajectories, which performs well in short horizon. For a longer prediction horizon over 3 seconds, vehicles’ intentions are important and have to be clarified first. Prototype classification [13] and intention recognition [14] are two research directions. Through collecting and clustering of tremendous trajectories in advance, prototype classification methods match online history trajectories with that in database. Multimodal future trajectories can be generated once matched. Intention recognition methods classify drivers’ potential maneuvers into limited classes. Combining heuristic information including road topology, traffic signal and vehicle turn signal, classification models such as Support Vector Machines (SVMs) [15] and Hidden Markov Models (HMMs) [16] are introduced to recognize surrounding drivers’ intentions. However, drivers’ behaviours are affected by various factors and highly personalized [17], which makes it hard to be recognized accurately in real application. Besides, a vehicle’s trajectory is not determined by its driver’s intention only but affected interactively.

To make an accurate prediction, it’s necessary to consider vehicles’ motion as a dynamic interactive system and model interactive impact among vehicles. Pairwise interactive modelling is of high complexity. Deo Nachiket [18] simplified interactive factor as a cost function that penalizes vehicle collision to filter out collide future trajectory pairs. Data-driven approaches are another practical methods and have been dug extensively. In these approaches, vehicles’ sequential features are learned through Long-Short Term Memory Networks (LSTMs). To learn interactive factor, Convolutional Neural Networks (CNNs) or max-pooling module is introduced. Deo et al. [19] divided certain surrounding area into grid cells and use CNNs to model spatial relationship among vehicles. Li et al. [4] built an adjacent matrix of surrounding vehicles where each elements represents pairwise proximity. A CNN is utilized to learn interactive factor. Messaoud et al. [20] discretized traffic environment into 3D grid cells and a Relational Recurrent Neural Network (RRNN) is used to predict future trajectory. Gupta et al. [21] utilized a pooling module to extract dense interactive features among vehicles. More recent works urge to merge data-driven and knowledge-driven approaches into a unified neural network [6], [7].

However, although prediction precision is improved continually, generalization and adaptation of proposed models still remain an open problem. Majority of research works validate proposed methods on one open dataset only. Some works validate on more datasets but train and test models for each dataset individually, which is not consistent with real application. Pushing forward to real application, a novel prediction model is urged to be proposed to cope with various traffic circumstances. Two questions arise naturally. How to measure difference among heterogeneous circumstances? And how to fulfill consistent vehicle trajectory prediction performance on emerging circumstances and visited ones, which is also called lifelong learning?

Divergence measurement of heterogeneous traffic circumstances is an open problem. To measure trajectory similarity, various methods are proposed, such as Euclidean distance (ED) [22], dynamic time warping (DTW) [23], longest common subsequence (LCSS) [24], merge distance (MD) [25], and spatiotemporal locality in-between polylines (STLIP) distance [26], etc. Su et al. [27] made a survey on 15 widely used trajectory distance measures in the literature. It can be deduced that ED is suitable for vehicle trajectories distance measurement that have the same total length and sample frequency. However, these similarity measurement methods consider one trajectory with another trajectory each time, while we aim to measure differences of traffic circumstances where dynamic number of trajectories are presented. In different traffic circumstances, vehicles’ interaction pattern changes and the way affecting future motion varies. In other words,
spatiotemporal dependency between future and past motion differs, which is essentially a conditional probability density function (CPDF) alteration problem. Therefore, a more reasonable method is to estimate distance between two unknown CPDF with empirical samples only.

Estimation distance of two unknown CPDF with samples only is challenging. As a commonly used probability divergence measurement method, KL divergence cannot work without analytic CPDF. The Donsker-Varadhan variational formula [28], [29] can be utilized to estimate CKLD empirically. However, it suffers from convergence problem for large divergence between two CPDFs, which is usually the case in real data. K nearest neighbor [30] is another approach to approximate CKLD. It requires distance calculation of condition data for two datasets, which is not committed for traffic circumstances with dynamic number of vehicles. As a conditional extension of traditional Maximum Mean Discrepancy (MMD) [31], conditional MMD (CMMD) [32] is proposed to measure distance between two CPDFs.

Similar with MMD, CMMD measures embedding probabilities distance in reproducing kernel Hilbert space (RKHS).

In MMD, a PDF is mapped into a point in RKHS, while in CMMD a CPDF is a family of points with different conditions. Therefore, CMMD is averaged for distances with different condition, which implies samples conditioned on a same condition. For two traffic circumstances in our work, condition data cannot guaranteed to be the same. In fact, CMMD is usually used as training loss function [33], [34] of neural networks where predicted and real value can be obtained on the same condition. Another method to measure PDF distance empirically is optimal transport. Tabak et al. [35] proposes a data-driven conditional optimal transport (COT) method. The COT represents empirical CPDF distance computation as a optimal transport problem constrained by CPDF alignment. CKLD is utilized to interpret the constraint and then converted into KL divergence between joint distributions through chain rule [36]. By using Donsker-Varadhan variational formula and Lagrange multiplier, the constrained COT is relaxed into a minimax optimization problem that can be optimized empirically. However, the COT is prone to local minimum and the minimax game is hard to converge. Moreover, for high dimension data as in our work, it is difficult to identify local minimum.

Lifelong learning aims to solve a series of tasks incrementally [37]. When addressing a new task, small amount or none data of old tasks are stored. After the final task is presented, all tasks should be solved by one task model with good performance. Key to lifelong learning is avoiding catastrophic forgetting of old tasks’ knowledge when updating the task model to solve a new task. From an aspect of model training, approaches to mitigate catastrophic forgetting are classified into three categories, architectural, regularization and rehearsal strategies. Architectural strategies train different models or subnetworks for incoming new tasks. A selector is used to choose an appropriate model or subnetwork for a task. Typical research works includes Progressive Neural Network (PNN) [38], Incremental Learning through Deep Adaptation (DAN) [39], Copy Weight with Re-init (CWR) [40], etc. These methods preserve performances of old tasks while conflicting with storage limitation of lifelong learning. Regularization strategies extend loss functions with additional term to retain performances of old tasks. Learning without Forgetting (LwF) [41] and Elastic Weight Consolidation (EWC) [42] are two representative methods. LwF proposes to use outputs of old models as soft targets to substitute data of old tasks, which is reported to suffer a buildup drop in old tasks’ performance as the task sequence grows longer [43]. EwC evaluates importance of parameters for old tasks and adds a penalty to changes when training on new tasks, which pays more attention to preserving the knowledge on old tasks but prevents the model from achieving competitive performance on new tasks [44]. Rehearsal strategies generally use an external memory to store part of old data [45], [46] or patterns [47]. As storage is limited and Generative Adversarial Networks (GANs) develops, generative replay [48] is proposed as a memory of previous data and its feasibility has been validate on several works [49], [50], [51], [52], [53]. Although quality of the generation model is a bottleneck, many works [49], [52], [54], [55], [56], [57] have proved that an elaborately designed generation model practically outperforms mainstream lifelong methods such as EWC, LwF, MAS [58], PathNet [59], and iCaRL [46] et al.

In this paper, CPDF distance between two traffic circumstances are calculated first to reveal spatiotemporal dependency divergence. Then a generative replay based lifelong trajectory prediction framework is proposed to enhance generalization and adaptation over different traffic environment. As a key of alleviate catastrophic forgetting, a generation model is realized through a novel conditional GAN (CGAN), which is called Recurrent Regression GAN (R2GAN). Through merging different generation models trained on different tasks, the generation model finally learns all knowledge involved in processed tasks. Eventually, a trajectory prediction model trained on generated data performs well on all tasks. A task chain including five tasks that stem from four open datasets is used as lifelong setup. Experiments on the task chain demonstrate effectiveness of proposed framework.

The rest of this paper is organized as follows. In section III, a mathematic formulation for lifelong trajectory prediction is addressed and overall framework is introduced. Divergence between two traffic circumstances is measured first in section IV. Then generative replay based lifelong prediction framework is introduced in section V in detail. In section VI, evaluation experiments are performed to evaluate quantitatively. Finally, conclusion and future work are introduced in section VI.

III. Problem Formulation

Formally, lifelong vehicle trajectory prediction is characterized by a set of tasks \( D = \{ d_1, d_2, \ldots, d_n \} \) to be learned by a parameterized model. In this context, tasks are specialized with vehicle trajectory prediction under different cities, environments at different time. For different tasks, individual datasets should be created to
ease the learning procedure, even though they are included in one large dataset. As lifelong learning represents a family of methods that accumulate knowledge and learn continuously with data available in sequential order [60], tasks involving different reasoning processes is out of the scope of lifelong learning. In addition, lifelong learning can learn continuously on consecutive tasks and keep considerable scope of lifelong learning. In addition, lifelong learning can ease the learning procedure, even though they are included in one large dataset. As lifelong learning represents a family of methods that accumulate knowledge and learn continuously with data available in sequential order [60], tasks involving different reasoning processes is out of the scope of lifelong learning.

In this work, with unsupervised learning nature of trajectory prediction, task data \( d_t \in D \) have training samples \( X_{t,t}^i \), where \( t = t_h + t_f \). Target vehicle and surrounding vehicles’ trajectories lasting for \( t_h \) are regarded as history information to train a parameterized model to predict future \( t_f \) trajectory of the target vehicle. In a traffic circumstance involving \( n_d \) vehicles, spatiotemporal dependency is formulated as a CPDF \( p(Y|X) \) where \( Y \) represents future \( t_f \) trajectory of the target vehicle and \( X \) represents history \( t_h \) trajectories of all vehicles. Samples are drawn i.i.d from an unknown distribution \( X_{t,t}^i \in P_{d_t} \) associated with task \( d_t \). Distribution \( P_{d_t} \) can be either different from or almost the same as another one. As a consequence, it is essential to quantify divergence among different tasks. In lifelong learning, task data \( d_t \) are observed sequentially and when the next data \( d_{t+1} \) arrive, data \( d_t \) are abandoned completely of only kept partly in a limited storage. Despite task divergence, the prediction model can predict accurately in all \( n \) tasks after observing all task data ultimately.

To address these two issues, a divergence measurement approach and a lifelong trajectory prediction framework are proposed as in Fig.1. CKLD is proposed to measure divergence of different driving circumstances and is not merged to a lifelong prediction framework. The core of CKLD is a network named MDN that extracts sequential feature of target vehicle’s history and spatial interaction feature between surrounding vehicles to estimate GMM parameters. Despite the existing divergence, a lifelong prediction framework is proposed that includes a trajectory generation model R2GAN and a trajectory prediction model. R2GAN is used to generate segments of trajectories, which is used as inputs to train a prediction model. In R2GAN, a regression discriminator extracts joint features including sequential feature of target vehicle’s history trajectory, interaction feature between target vehicle and surrounding vehicles, and sequential feature of target vehicle’s future trajectory. As the R2GAN memorizes all experienced circumstances, through encoding and decoding sequential feature of target vehicle’s history and spatial interaction feature between surrounding vehicles, the trained prediction model can achieve good prediction performance on all visited circumstances consistently.

### IV. DIVERGENCE MEASUREMENT OF DIFFERENT TRAFFIC CIRCUMSTANCES

As an effective divergence measurement method, KL divergence is extended to CKLD to measure spatiotemporal dependency difference of two traffic circumstances, which is formulated as

\[
CKLD(p_1(Y|X)||p_2(Y|X)) = \int p_1(X) \left( \log \frac{p_1(Y|X)}{p_2(Y|X)} \right) p_1(Y|X) dYdX. \tag{1}
\]

The CKLD can not be computed without analytic formulation of \( p(Y|X) \). Therefore, parameters of GMMs are estimated by a MDN to approximate \( p(Y|X) \) first and then MC sampling can be performed to calculate CKLD.

#### A. Dimension Normalization for Dynamic Traffic Circumstances

In a dynamic traffic circumstances involving \( n_d \) vehicles, condition \( X \) should be represented as \( X = (x_1^{t_h}, x_2^{t_h}, \ldots, x_n^{t_h}) \), where \( x_i^{t_h} \) represents sequential coordinate of vehicle \( i \) that lasts \( t_h \), which possesses dynamic dimension. To facilitate model learning, a fixed dimension is preferred. Notice that target vehicle’s future motion is affected by limited number of neighboring vehicles, it is reasonable to consider \( n_v \) closest vehicles only, which is also a common practice in trajectory prediction research [4], [61]. To represent interactive relationship between considered vehicles, a Laplacian matrix is calculated. Being different from usual 3D case [62], a 2D Laplacian matrix is calculated through weighting on time dimension. Then eigenvectors corresponding to the biggest \( k \) eigenvalues are concatenated with target vehicle’s history trajectory, which forms a condition vector \( X = (x_1^{t_h}, v_1, \ldots, v_k) \) with fixed dimension \( d_X = 2t_h + kn_v \), where the superscript \( e \) represents the target vehicle. The Laplacian matrix is calculated through

\[
L = D - A, \\
A = (a_{ij})_{n_v \times n_v}, \\
D = (d_{ij})_{n_v \times n_v}, \\
a_{ij} = \exp \left( -\sum_{k=1}^{t_h} w_k d \left( x_i^k, x_j^k \right) / \sum_{k=1}^{t_h} w_k \right), \\
w_k = \lambda^{t_h-k}, \quad k = 1, \ldots, t_h, \\
d_{ij} = \begin{cases} 
\frac{1}{2} \sum_{j=1}^{n_v} d_{ij}, & i = j \\
0, & i \neq j.
\end{cases}
\tag{2}
\]

![Fig. 1. Overall framework.](image-url)
where \( d(x^t_k, x^t_j) \) is ED between vehicle \( i \) and \( j \) at time \( k \) and \( \lambda \) is a decay parameter.

### B. Estimation on GMMs Based on a MDN

To calculate CKLD between two CPDFs, GMMs are introduced to approximate CPDF as

\[
p(Y \mid X) = \sum_{i=1}^{m} \alpha_i(X) \phi_i(Y \mid X),
\]

where \( m \) is number of Gaussian distribution hypothesis and \( \phi_i(Y \mid X) = \exp \left( -\frac{\|Y - \mu_i(X)\|^2}{2\sigma_i(X)^2} \right) / (2\pi)^{d/2} \sigma_i(X)^d \). For \( i = 1, \ldots, m \), mixing coefficient \( \alpha_i(X) \), mean \( \mu_i(X) \), and variance \( \sigma_i(X) \) are estimated through MDN [63], [64]. As shown in Fig.2, a Multi-Layer Perceptron (MLP) is applied for input \( X \) to obtain a feature encoding \( Z \). Then three separate Fully Connected (FC) layers are utilized to calculate parameters of GMMs. To enforce \( \sum_{i=1}^{m} \alpha_i(x) = 1 \), a softmax function is applied \( \alpha_i(X) = \exp(FC(Z_i)) / \sum_{j=1}^{m} \exp(FC(Z_j)) \), where \( FC(\bullet) \) represents a FC layer and the subscript \( i \) and \( j \) represents vector component. Means are unconstrained. Variances \( \sigma_i(X) \) should be positive. A softplus function is applied hence \( \sigma_i(X) = \log (1 + \exp (FC(Z_i))) \). Training loss function for MDN is \( L_{mdn} = -\log p(Y \mid X) \).

### C. Calculation of CKLD Through Monte-Carlo Sampling

After GMMs are estimated for each condition \( X_i \), CKLD can be computed. As in (1), for every sample condition \( X_i \), \( i = 1, \ldots, n_1 \) on \( p_1(X) \), CKLD can be calculated as

\[
KLD\left( p_1(Y \mid X_i) \parallel p_2(Y \mid X_i) \right) = \int \log \left( \frac{p_1(Y \mid X_i)}{p_2(Y \mid X_i)} \right) p_1(Y \mid X_i) dY.
\]

Although KL divergence between two GMMs is not analytically attractable, some techniques are developed to estimate effectively. Hershey [65] compared 5 methods and concluded that MC sampling reaches clearly the best accuracy. Suppose samples \( Y_j, j = 1, \ldots, n_{mc} \) are sampled from \( p_1(Y \mid X_i) \), then CKLD can be calculated as

\[
KLD\left( p_1(Y \mid X_i) \parallel p_2(Y \mid X_i) \right) = \sum_{j=1}^{n_{mc}} \left( \log p_1(Y_j \mid X_i) - \log p_2(Y_j \mid X_i) \right) / n_{mc}.
\]

CKLD can be calculated as

\[
CKLD\left( p_1(Y \mid X) \parallel p_2(Y \mid X) \right) = \sum_{i=1}^{n_1} KLD\left( p_1(Y \mid X_i) \parallel p_2(Y \mid X_i) \right) / n_1.
\]

The complete CKLD computation flow is summarized in Algorithm 1.

### Algorithm 1 CKLD Between Two Traffic Circumstances

**Require:** Sample pairs \( (X_{1i}, Y_{1i}) \sim p_1(X, Y), i = 1, \ldots, n_1 \) and \( (X_{2i}, Y_{2i}) \sim p_2(X, Y), i = 1, \ldots, n_2 \).

**Ensure:** \( CKLD(p_1(Y \mid X) || p_2(Y \mid X)) \).

1. Calculate Laplacian matrix according to (2) and normalize condition to uniform dimension.
2. For \( i = 1, \ldots, n_1 \)
   - Fit a MDN with \( (X_{1i}, Y_{1i}) \) and loss function \( L_{mdn} \).
3. For \( i = 1, \ldots, n_2 \)
   - Fit a MDN with \( (X_{2i}, Y_{2i}) \) and loss function \( L_{mdn} \).
4. For \( CKLD \leftarrow 0 \)
   - For \( i = 1, \ldots, n_1 \)
     - Sampling \( Y_j \sim p_1(Y \mid X_i), j = 1, \ldots, n_{mc} \)
     - Calculate KLD according to (4), \( CKLD \leftarrow CKLD + KLD(p_1(Y \mid X_i) \parallel p_2(Y \mid X_i)) \).
5. For \( CKLD \leftarrow CKLD / n_1 \).
6. Return \( CKLD \).

### V. GENERATIVE REPLAY BASED LIFELONG TRAJECTORY PREDICTION

Based on our previous research on trajectory generation [66], a novel trajectory generation model trained by the standard GAN [67] loss is proposed to memorize data distribution of tasks. With a vehicle trajectory prediction model trained by generated trajectories, a lifelong vehicle trajectory prediction framework is realized.

It is worth emphasizing that it is not a generative prediction model. A generation prediction model takes vehicles’ history model as generator’s inputs and takes target vehicle’s future trajectory and predicted trajectory as discriminator’s inputs to discriminate between them. Therefore, a generative prediction model tries to build a generator that learns mapping between history and future distribution. Nevertheless, it is not capable of learning continuously. While in our lifelong learning framework, with a specialized learning and merging schema, a generation model is utilized to memorize all experienced scenario. The replayed trajectories including history and future trajectories both are used to train a trajectory prediction model later on. As an alternative, a generative prediction model can replace the trajectory prediction model used in our framework to constitute a lifelong trajectory prediction model.

#### A. Generator Conditioned on Relative Position Configuration

In a vehicle trajectory prediction task, we need to predict a target vehicle’s future \( f_T \) trajectory according to its \( h_T \) history trajectory and its neighboring vehicles’ histories. To facilitate generation process, a full prediction scenario is required to be generated that consists of target vehicle’s and its neighboring vehicles’ trajectories lasting for \( t = h_T + f_T \) horizon.

Being different from traditional GANs that model generation procedure as mapping from a prior probability distribution to a target one, we are inspired by Quant GAN [68] and model prediction scenario generation as
map between stochastic processes. Gaussian process with RBF kernel is selected as prior stochastic process. For single vehicle, Gaussian process samplings are obtained, which constitutes \((N_1^1, N_2^1, \ldots, N_n^1)\) for total \(n\) interactive vehicles. Conditional GANs are easier to train and make generated samples more controllable. Therefore, multiple vehicles’ spatial configuration condition \((C_1, C_2, \ldots, C_n)\) is utilized as conditional inputs for our generator, where \(C_i\) represents condition for vehicle \(i\). Spatial configuration condition \((C_1, C_2, \ldots, C_n)\) and input sampling \((N_1^1, N_2^1, \ldots, N_n^m)\) are encoded by two MLPs individually first. To map sequential feature of Gaussian process into target data, a bidirectional GRU is utilized, where initial hidden states \(h_{g0}\) and \(h_{gb0}\) are set by encoded position condition [69], [70]. In the bidirectional GRU, forward code \(E_{1,1}^j\) of agent \(j\) and backward code \(E_{b1,1}^j\) are averaged to form sequential code \(E_{1,1}^j\). A MLP is attached later to encode spatial relations. A fully connected (FC) layer is used later with tanh activation function to output spatiotemporal data \((\tilde{X}_1^1, \tilde{X}_2^1, \ldots, \tilde{X}_n^1)\) of \(n\) agents. Complete framework is shown in Fig.3.

Let \(G(\bullet)\) represents a GRU and \(M(\bullet)\) a MLP unit, the generation procedure can be formulated as:

\[
E_{1,1}^j = G_{gf} \left( \frac{M_{g0}(N_1^j - m, N_m^j - m + 1, \ldots, N_n^j)}{h_{g0} = M_{e1} (C_j)}, h_{gb0} = M_{e2} (C_j) \right), \]
\[
E_{b1,1}^j = G_{gb} \left( \frac{M_{g0}(N_1^j - m, N_m^j - m + 1, \ldots, N_n^j)}{h_{gb0} = M_{e2} (C_j)} \right), \]
\[
E_{1,1}^j = M_{g3} \left( \frac{E_{1,1}^j + E_{b1,1}^j}{2} \right), \]
\[
E_{1,1}^j = E_{1,1}^j - E_{1,1}^j, \]
\[
\tilde{X}_1^1 = M_{g5} \left( E_{1,1}^j, \sum_{i=1, i \neq j}^n M_{g4} \left( E_{1,1}^j \right) \right) / (n - 1), \]
\]

for agent \(i, j = 1, 2, \ldots, n\).

**B. MLP Based Regression Discriminator**

As generated prediction scenario data are used for vehicle trajectory prediction task, it is of significant importance to maintain sequential dependences in generated samples. For this, a regression discriminator is first proposed by us to distinguish multiple agents’ real data from generated one. The regression discriminator learns to model joint distribution of inputs and outputs in a prediction task. Specifically, distribution of target vehicle’s history data, target vehicle’s future data and neighboring vehicles’ history data are taken as inputs of the regression discriminator and modelled jointly. The regression discriminator outputs a classification probability that indicates degree of true.

Architecture of a regression discriminator is shown in Fig.4. For vehicle \(j = 1, 2, \ldots, n\), trajectory data \(\tilde{X}_1^j\) are pre-processed through

\[
\tilde{X}_1^j = \tilde{X}_1^j - \tilde{X}_1^c, \]
\[
X_1^j = \tilde{X}_1^j / \left\| \tilde{X}_1^j, \ldots, \tilde{X}_n^1 \right\|_\infty. \]

After centering and normalization pre-processing, target vehicle data are separated from neighboring agents’ data. A MLP is applied to the target vehicle to encode its history \(X_1^c\) and future data \(X_{b+1}^j + t_f \) into \(E_{rd}^j\) and \(E_{rd}\) specifically. Relative difference between the target vehicle and neighboring vehicles are calculated and encoded into \(E_{rd}^j\) through a MLP with a mean pooling layer, which is invariant to neighboring vehicles’ sequence. All codes are concatenated and encoded by two FC layers to get a feature vector \(F_{rd}\). Finally, another FC layer is applied to the feature vector to obtain classification probability \(L_{rd}\).

**Computation workflow can be formalized as:**

\[
E_{rd}^j = M_{rd0} \left( X_{1,b}^c \right), \]
\[
E_{rd} = M_{rd0} \left( X_{b+1,b+1}^c + t_f \right), \]
\[
E_{rd}^{net} = \left( \sum_{i=1}^n M_{rd1} \left( X_{c,b+1}^c \right) \right) / (n - 1), \]
\[
F_{rd} = FC_{rd0} \left( F_{rd}^h, F_{rd}^j, E_{rd}^{net} \right), \]
\[
L_{rd} = FC_{rd1} (F_{rd}). \]

**C. Evolution of Generation Model**

In a lifelong task chain \(D = \{d_1, \ldots, d_n\}\), generation models are required to be merged to a long-term model when a new task \(d_{i+1}\) arrives. In general, there are two fusion methods. As illustrated in Fig.5, one method [48], [50] merges generation model \(G_t\) trained by task \(d_i\) with the new task \(d_{i+1}\). Generated samples from long-term model \(G_t\) and real samples drawn from \(d_{i+1}\) are combined as real samples to train a new generation model \(G_{i+1}\), which we call Long-term-Merge(DGM-LTM) method. Another method [49], [55] trains a temporal generation model \(G_{i+1}\) for the new task \(d_{i+1}\). Generated samples from long-term model \(G_t\) and temporal model \(G_{i+1}\) are combined as real samples to train a new long-term model \(G_{i+1}\), which we call Long-term-Merge(LTM) method. Although LDM performs better than LTM method intuitively, they are both applied to our lifelong task and compared.

**D. Task Model for Vehicle Trajectory Prediction**

To perform vehicle trajectory prediction task with generated prediction scenario, a prediction model is proposed. As with mainstream trajectory prediction methods, history information and interactive relationship between target vehicle and neighboring vehicles are utilized to predict future trajectory of the target vehicle. The key component representing sequential invariant interaction feature is a mean pooling module [21]. Overall architecture is shown in Fig.6. First, target vehicle’s trajectory \(X_1^c\) is separated from neighboring vehicles and encoded by a LSTM layer. Then, difference between history trajectories of target vehicle and others are encoded by a MLP and a mean pooling layer. These two parts are concatenated and encoded by a MLP further. A LSTM layer and a MLP is used to output predicted trajectories \(\tilde{X}_{b+1,b+1}^c\).
Let $LSTM(\cdot)$ represents a LSTM unit, then for a prediction workflow, we have

$$E^h_r = M_r \left( LSTM(M_{r0} \left( X^e_{1:t_h} \right) ) \right),$$
$$E_{rel}^{nei} = M_r \left( \sum_{i=1}^{n} M_{r1} \left( X^e_{i:t_h} \neq e \right) \right),$$
$$\hat{X}_{b_h+1:t_h+f}^e = M_{r3} \left( LSTM \left( E^h_r, E_{rel}^{nei} \right) \right). \quad (8)$$

With a merged generation model through LDM or LTM method and a prediction model, a generative replay based vehicle prediction model GRTP-D or GRTP-T is realized by training the prediction model through generated trajectories. As the generation model memorizes all past experiences, the GRTP-D or GRTP-T model is expected to realize considerable performance on all tasks. It should be noted that although the backbone network of the generation model looks similar with the prediction model, guiding the prediction model is not trivial or even not practical. In the generation model, the generator learns spatiotemporal relationship transformation between Gaussian process samplings and vehicle trajectories, while spatiotemporal dependency between history and future trajectories are learned by a prediction model. The discriminator in the generation model can be regarded as a classifier, which is also hard to deduce a classification model [71].

VI. EXPERIMENTS AND ANALYSES

All experiments are realized via Pytorch [72]. Running environment is Ubuntu 16.04, Intel Core i9-9900X CPU, GeForce GTX 1080 Ti, and 64GB RAM. All code including CKLD calculation and lifelong learning experiments and pre-processed data are available online.\(^1\)

A. Dataset and Generation Model Setup

Following common lifelong learning evaluation setup where several related datasets are selected to train and test consecutively [37], [41], [49], five sub-datasets recorded in different locations are selected from four open datasets. Some traffic circumstances are illustrated in Fig. 7.

- **NGSIM dataset [73].** The NGSIM dataset contains two sub-datasets, US101 dataset and I801 dataset that are recorded on southbound US 101 and eastbound I-80 specifically. As tremendous vehicle trajectories are recorded, it’s time consuming to learn them all. Therefore, 7:50 a.m. to 8:05 a.m. trajectory records in US101 dataset are selected and named with $US_{101-1}$. To simulate a case happened in real application where drives visit same place at different time period, 8:05 a.m. to 8:20 a.m. trajectory records in US101 dataset are selected as $d_5$ and named with $US_{101-2}$. Without losing generality, we keep full prediction scenarios that contains 4 and 5 vehicles only to ease the learning burden furtherly and over 188k items still remains. I801 dataset are refined as the same way and over 129k items remains. The selected datasets are separated into training, validation and testing dataset by 7:1:2.

- **HighD dataset [74].** The highD dataset is a new dataset of naturalistic vehicle trajectories recorded at six different locations on Germany highways, which results in sixty recordings. Considering learning burden, the 20th recording is selected and pre-processed, which results in 88k items left for training. It is noted that as trajectories are recorded in 10HZ in NGSIM while 25HZ in HighD dataset, we aim to generate trajectories in 5HZ. As recording items is not comparable with NGSIM dataset in scale, full prediction scenarios that contains 2 to 9 vehicles are kept. For ease of representation, this dataset is called $highd_{20}$.

- **Interaction dataset [75].** The interaction dataset contains naturalistic motions of various traffic participants in a

\(^1\)https://github.com/CliffBao/GRTP
variety of highly interactive driving scenarios from different countries. Trajectories in DR_CHN_Merging_ZS map is a lane merging dataset in China urban area. To be comparable with other datasets in scale, five trajectories records in the map are merged into a dataset named \texttt{inter5d}, which consists of 126k items.

Therefore, a lifelong task chain is formed by above five datasets

\[
D = \{d_1, d_2, d_3, d_4, d_5\} = \{\text{US101-1, i801, highd20, inter5d, US101-2}\}. \quad (9)
\]

B. Divergence Between Datasets

To measure CKLD between two datasets, we fix maximum vehicle number to \(n_v = 5\) and only top \(k = 3\) eigenvectors are extracted. Gaussian hypothesis number of GMMs is set into \(m = 20\). The MDN is optimized through adam [76] optimizer with learning rate \(\gamma_m = 0.0004\) and batch size \(b_m = 4096\). CKLD results of pairwise datasets are presented in Table I. In Table I, CKLD between \(d_1\) and \(d_5\) is the

| CKLD   | dataset 2 | \(d_1\) | \(d_2\) | \(d_3\) | \(d_4\) | \(d_5\) |
|--------|-----------|--------|--------|--------|--------|--------|
| \(d_1\) |           | 0      | 22.14  | 503.22 | 98.43  | 16.72  |
| \(d_2\) |           | 29.00  | 0      | 756.03 | 98.57  | 24.44  |
| \(d_3\) |           | 88.46  | 53.87  | 0      | 142.26 | 121.42 |
| \(d_4\) |           | 75.05  | 63.54  | 652.00 | 0      | 62.50  |
| \(d_5\) |           | 19.57  | 17.09  | 511.17 | 93.49  | 0      |

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
closest, which is expected as they are collected in the same highway at different time period. Divergence between $d_3$ and other datasets are quite large because only $d_3$ is collected in Germany. The same goes for $d_4$.

Another annotation is about robustness to dataset quality. CKLD calculates the divergence of conditional distribution between two datasets. Therefore, in theory, CKLD will not be affected by some errors significantly. From Table I, it can be concluded that quantitative results are consistent with our expectation, which demonstrates the robustness to data quality.

### C. Validation of Trajectory Prediction Model

We compare proposed the task model performance with other benchmark trajectory prediction models.

- **Constant Velocity (CV)**. A constant velocity Kalman filter reported in [19].
- **GAIL-GRU**. A generative adversarial imitation learning model described in [77].
- **LSTM with fully connected social pooling (S-LSTM)**. This uses the fully connected social pooling described in [78] and generates a unimodal output distribution.
- **LSTM with convolutional social pooling (CS-LSTM)**. This uses convolutional social pooling and generates a unimodal output distribution [19]. As designing an extraordinary prediction model is not our main goal, we use results reported in [19].

To present a fair comparison, full NGSIM dataset is used to demonstrate validity of proposed prediction model, which is different from lifelong setups. RMSE performance for varied prediction horizon ($PH$) is given out in Table II. Suppose $BS$ batch size predicted future trajectories $\tilde{X}_i^t$, $i = 1, 2, \ldots, BS$ at time $t$ are calculated and real future trajectories $X_i^t$, $i = 1, 2, \ldots, BS$ are available, then RMSE at time $t$ is

$$RMSE(t) = \sqrt{\frac{\sum_{i=1}^{BS} (X_i^t - \tilde{X}_i^t)^2}{BS}}.$$  \hfill (10)

From Table II, it can be concluded that the task model possess similar prediction ability with mainstream methods. As we are not aiming to significantly improve prediction accuracy on single dataset but instead improve generality and lifelong prediction ability over multiple tasks, the performance of proposed prediction model may not compatible with state-of-the-art methods.

### D. Evaluation of Generation Model

Evaluation of sequential-like generation model lacks standard benchmark, especially for spatiotemporal data. Considering that sequential data are commonly used for prediction, we adopt TRTR and TSTR method to evaluate performance of our generation model R2GAN, which is applied widely for quality evaluation of time-series data [66], [79], [80]. TRTR means to train a task model with real samples and test it on real samples to solve a prediction task. Performance of the task model is regarded as the best performance. TSTR means to train a task model with synthetic samples while test it on real samples, which simulates real application procedure of a generation model and evaluates applicability of generated samples. The closer between TSTR and TRTR performance, the better the generation model.

To evaluate effectiveness of proposed R2GAN model, comparison experiments between TRTR and TSTR are conducted on five datasets respectively. Take $d_1$ as an example. In TRTR, a subset of $d_1$ is used to train a prediction model that is tested on another subset of $d_1$. In TSTR, the same training subset of $d_1$ is used to train a R2GAN. Generated trajectories from the trained R2GAN is used to train another prediction model that is tested on the same test subset of $d_1$ as in TRTR. RMSE performance are given out in Table III. From Table III, it is obvious that TSTR performance is comparable with TRTR. In some cases, TSTR even obtains minor advantages over TRTR, which tends to result from some randomness during model training and testing. As a result, it is demonstrated sufficiently that the proposed R2GAN memorizes essential spatiotemporal features that lie in vehicle trajectories.

### E. Lifelong Trajectory Prediction

In R2GAN, a generator takes inputs as several Gaussian process samplings and label conditions of trajectories that indicate relative position to a target vehicle. For example, $-1$ indicates a vehicle is located on the left lane of the target vehicle at the beginning. The generator outputs corresponding vehicle trajectory snippets lasting for 8 seconds, i.e. 41 steps. For a regression discriminator, real or generated prediction scenario are classified as real or fake. R2GAN is trained by adam [76] optimizer with learning rate $\gamma = 0.0001$. Non-saturating GAN loss function is applied as in [81]. To demonstrate lifelong prediction ability of proposed framework, three other methods are realized and compared with our approach.

### TABLE II

| PH(s) | 1    | 2    | 3    | 4    | 5    |
|-------|------|------|------|------|------|
| CV    | 0.73 | 1.78 | 3.13 | 4.78 | 6.68 |
| GAIL-GRU | 0.69 | 1.51 | 2.55 | 3.65 | 4.71 |
| S-LSTM | 0.65 | 1.31 | 2.16 | 3.25 | 4.55 |
| CS-LSTM | 0.61 | 1.27 | 2.09 | 3.10 | 4.37 |
| Ours  | 0.55 | 1.28 | 2.18 | 3.30 | 4.64 |

### TABLE III

| PH(s) | 1    | 2    | 3    | 4    | 5    |
|-------|------|------|------|------|------|
| $d_1$ | TRTR | 0.79 | 1.79 | 3.15 | 5.01 | 7.30 |
| TSTR  | 0.82 | 1.89 | 3.26 | 5.03 | 7.24 |
| $d_2$ | TRTR | 0.90 | 1.90 | 3.06 | 4.43 | 6.08 |
| TSTR  | 0.89 | 1.89 | 3.07 | 4.52 | 6.17 |
| $d_3$ | TRTR | 0.52 | 1.21 | 2.02 | 2.90 | 4.24 |
| TSTR  | 0.69 | 1.43 | 2.27 | 3.30 | 4.34 |
| $d_4$ | TRTR | 0.49 | 1.13 | 2.01 | 3.09 | 4.38 |
| TSTR  | 0.43 | 1.00 | 1.90 | 2.97 | 4.16 |
| $d_5$ | TRTR | 0.60 | 1.43 | 2.55 | 3.93 | 5.55 |
| TSTR  | 0.57 | 1.42 | 2.56 | 4.00 | 5.76 |
Generative replay based trajectory prediction (GRTP). This is the proposed lifelong trajectory prediction model based on generative replay. Resulted from two fusion methods LDM and LTM, the GRTP is furtherly classified into GRTP-D and GRTP-T specifically.

- Joint training (JT). Joint training violates essential storage limitation and assumes that all data are available. This is regarded as the best possible performance over any lifelong learning methods.
- Fixed model (FM). A trajectory model trained by task $d_1$ will not be adjusted anymore and will be applied to new tasks directly.
- Fine tuning (FT). A trajectory model is trained while new task data $d_i$ are available. This is a possible choice but is expected to forget everything about old tasks. From some perspective, research works on trajectory prediction model design and optimization can be categorized into this method, although they did not test the model trained on new dataset on old ones.

The RMSE plots through the full lifelong task chain is illustrated in Fig. 8, where local RMSE around $t_f$ is zoomed in by a mini plot. As the lifelong task chain proceeds from $d_1$ to $d_5$, prediction performance on future 5 seconds horizon is validated on more tasks. Exact numeric result after addressing $d_5$ is given out in Table IV. From experiment results, we can see that

- the JT performs best in $d_4$ and part of $d_2$ instead of all tasks, which can be caused by parameter tuning during training or local convergence minimum. Another possible reason is joint training learns average minimum loss over all datasets. As a consequence, average performance on all five datasets rather than individual component datasets would be fairer to joint training. The average results in Table IV demonstrate our analysis.
- It is obvious FT forgets old knowledge while attaining new knowledge. In Table IV, FT is trained consecutively by five datasets, which means the latest training dataset is $d_5$. Indeed, FT is a common practice when new task arrives and performs well if divergence between old and new task is small. From CKLD computation result, divergence between $d_1$, $d_2$, and $d_3$ is relatively small. Therefore, FT performs well on these three tasks after $d_5$. On the contrary, CKLD between $d_5$ and $d_3$, $d_4$ is

| $d_1$ | JT  | FM  | FT  | GRTP-D | GRTP-T |
|-------|-----|-----|-----|--------|--------|
| RMSE  | 0.79| 1.63| 2.86| 0.80   | 0.81   |
| $d_2$ | JT  | FM  | FT  | GRTP-D | GRTP-T |
| RMSE  | 1.09| 2.09| 3.48| 1.00   | 0.99   |
| $d_3$ | JT  | FM  | FT  | GRTP-D | GRTP-T |
| RMSE  | 1.46| 1.66| 2.97| 1.00   | 0.99   |
| $d_4$ | JT  | FM  | FT  | GRTP-D | GRTP-T |
| RMSE  | 1.36| 3.30| 6.06| 1.00   | 0.99   |
| $d_5$ | JT  | FM  | FT  | GRTP-D | GRTP-T |
| RMSE  | 0.71| 1.32| 2.30| 0.81   | 0.88   |

### Table IV: RMSE(M) Comparison of Different Models After Finishing Lifelong Task Chain

| PH(ε) | 1    | 2    | 3    | 4    | 5    |
|-------|------|------|------|------|------|
| JT    | 0.93 | 1.86 | 3.30 | 5.14 | 7.30 |
| FM    | 0.79 | 1.82 | 3.18 | 4.94 | 7.06 |
| FT    | 0.79 | 1.63 | 2.86 | 0.49 | 6.53 |
| GRTP-D| 0.80 | 1.89 | 3.19 | 5.08 | 7.32 |
| GRTP-T| 0.81 | 1.86 | 3.29 | 5.08 | 7.28 |
| JT    | 1.09 | 2.15 | 3.33 | 5.09 | 6.80 |
| FM    | 1.08 | 2.56 | 4.54 | 6.91 | 9.57 |
| FT    | 1.09 | 2.09 | 3.48 | 2.56 | 7.34 |
| GRTP-D| 1.00 | 2.25 | 3.79 | 5.61 | 7.67 |
| GRTP-T| 0.99 | 2.10 | 3.49 | 4.97 | 6.85 |
| JT    | 1.46 | 1.66 | 2.97 | 4.94 | 7.01 |
| FM    | 1.36 | 3.30 | 6.06 | 9.24 | 12.25 |
| FT    | 1.32 | 3.21 | 6.49 | 10.01| 13.81|
| GRTP-D| 0.81 | 1.47 | 2.68 | 4.66 | 7.07 |
| GRTP-T| 0.70 | 1.88 | 3.41 | 4.93 | 6.66 |
| JT    | 0.36 | 0.96 | 1.89 | 2.95 | 4.09 |
| FM    | 0.44 | 1.28 | 2.34 | 3.55 | 4.86 |
| FT    | 0.79 | 1.32 | 2.30 | 3.69 | 5.33 |
| GRTP-D| 0.57 | 1.35 | 2.33 | 3.56 | 4.89 |
| GRTP-T| 0.47 | 1.25 | 2.25 | 3.46 | 4.82 |
| JT    | 0.71 | 1.48 | 2.70 | 4.21 | 5.92 |
| FM    | 0.65 | 1.52 | 2.65 | 4.07 | 5.67 |
| FT    | 0.62 | 1.31 | 2.32 | 3.63 | 5.20 |
| GRTP-D| 0.62 | 1.47 | 2.64 | 4.15 | 5.90 |
| GRTP-T| 0.63 | 1.54 | 2.75 | 4.24 | 5.96 |
| JT    | 0.90 | 1.34 | 3.10 | 4.61 | 6.39 |
| FM    | 0.93 | 2.22 | 3.97 | 6.08 | 8.38 |
| FT    | 1.52 | 2.86 | 4.57 | 6.65 | 9.43 |
| GRTP-D| 0.87 | 1.81 | 3.19 | 4.77 | 6.70 |
| GRTP-T| 0.85 | 1.89 | 3.23 | 4.85 | 6.76 |
relatively large, which results in poor performance on \(d_3\) and \(d_4\) after tuning on task \(d_5\). The same consequence of applying FT to new task is also remarkable in Fig.8.

- FM is trained on \(d_1\) only. As a result, good performance on \(d_1\) and \(d_2\) is attainable after \(d_3\) while large RMSE is observed on other tasks.

- The proposed GRTP-T and GRTP-D perform well consistently over all tasks and possess close RMSE to JT. As JT stores all task data and can be considered as the best possible performance in lifelong task chain, we can conclude that GRTP mitigates catastrophic forgetting and realizes lifelong learning whether with LDM or LTM fusion method.

- Although intuitively thinking, GRTP-D will outperforms GRTP-T as it merges long-term generation model with new data directly and avoids training a new generation model on new data, which avoids distribution learning bias introduced by the temporal generation model. However, it can be observed in Table IV and Fig.8 that no significant performance gap exists between them. This observation demonstrates that minor or even no distribution bias is introduced by our proposed R2GAN, which validates effectiveness of proposed R2GAN and model merging method implicitly.

VII. CONCLUSION

Maintaining consistent performance on vehicle trajectory prediction over different traffic circumstances is of significant importance for safe driving. Lifelong trajectory prediction is first addressed by us. Key problem hindering lifelong learning is catastrophic forgetting which arises from existence of spatiotemporal dependency divergence. To analysis the divergence between different traffic circumstances, CKLD is calculated based on GMMs approximation and MC sampling. Then a R2GAN is developed to generate dynamic number of vehicles in traffic circumstances, which guarantees inherent spatiotemporal dependency through a novel regression discriminator. Two methods are applied to construct a lifelong R2GAN model, LTM and LDM. LTM merges generated trajectories from long-term R2GAN and temporal R2GAN to train a new long-term generation model. LDM takes trajectories generated from long-term R2GAN and sampled from real dataset as real data to update the long-term generation model. Different spatiotemporal dependency are remembered by the long-term generation model that capable of generating samples from all processed tasks, thus mitigating catastrophic forgetting problem. Both merging method are validated through a constructed lifelong task chain and fulfill lifelong trajectory prediction task with consistent performance.

In this work, five tasks are applied to verify effectiveness of proposed lifelong framework. After building a feasible lifelong vehicle trajectory prediction framework, in future work, more diversified traffic circumstances can be introduced to improve lifelong prediction performance. To promote prediction precision, explicit map topology can also be generated along with matched vehicle trajectories. In addition, real time performance of lifelong learning and integrating is subsequent consideration. It should be noted that generative replay based lifelong framework is one of three possible frameworks. Regularization and architectural strategies are also potential foundations of new frameworks.

REFERENCES

[1] S. Qiao, N. Han, J. Wang, R.-H. Li, L. A. Gutierrez, and X. Wu, “Predicting long-term trajectories of connected vehicles via the prefix-projection technique,” IEEE Trans. Intell. Transp. Syst., vol. 19, no. 7, pp. 2305–2315, Jul. 2018.

[2] L. Hou, L. Xin, S. E. Li, B. Cheng, and W. Wang, “Interactive trajectory prediction of surrounding road users for autonomous driving using structural-LSTM network,” IEEE Trans. Intell. Transp. Syst., vol. 21, no. 11, pp. 4615–4625, Nov. 2020.

[3] J. Li, H. Ma, and M. Tomizuka, “Interaction-aware multi-agent tracking and probabilistic behavior prediction via adversarial learning,” in Proc. Int. Conf. Robot. Autom. (ICRA), May 2019, pp. 6658–6664.

[4] X. Li, X. Ying, and M. C. Chuaah, “GRP; Graph-based interaction-aware trajectory prediction,” in Proc. IEEE Intell. Transp. Syst. Conf. (ITSC), Oct. 2019, pp. 3960–3966.

[5] R. Chandra, U. Bhattacharya, A. Bera, and D. Manocha, “TraPhic: Trajectory prediction in dense and heterogeneous traffic using weighted interactions,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 8475–8484.

[6] A. Dulan and J. C. Murray, “Multi-modal anticipation of stochastic trajectories in a dynamic environment with conditional variational autoencoders,” 2021, arXiv:2103.03912.

[7] M. Bahari, I. Nejjar, and A. Alahi, “Injecting knowledge in data-driven vehicle trajectory predictors,” 2021, arXiv:2103.04854.

[8] A. Zyner, S. Worrall, and E. Nebot, “Naturalistic driver intention and path prediction using recurrent neural networks,” IEEE Trans. Intell. Transp. Syst., vol. 21, no. 4, pp. 1584–1594, Apr. 2020.

[9] J. Zhu, X. Lin, A. K. Jain, and J. Zhou, “Transfer learning in deep reinforcement learning: A survey,” 2020, arXiv:2009.07888.

[10] C. H. Lampert, H. Nickisch, and S. Harmeling, “Learning to detect unseen object classes by between-class attribute transfer,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2009, pp. 951–958.

[11] S. Ammoun and F. Nashashibi, “Real time trajectory prediction for collision risk estimation between vehicles,” in Proc. IEEE 5th Int. Conf. Intell. Comput. Commun. Process., Aug. 2009, pp. 417–420.

[12] P. Lytrivis, G. Thomaidis, and A. Amditis, “Cooperative path prediction in vehicular environments,” in Proc. 11th Int. IEEE Conf. Intell. Transp. Syst., Oct. 2008, pp. 803–808.

[13] Q. Tran and J. Firl, “Online maneuver recognition and multimodal trajectory prediction for intersection assistance using non-parametric regression,” in Proc. IEEE Trans. Intell. Vehicles Symp., Jun. 2014, pp. 918–923.

[14] G. S. Aouda, V. R. Desaraju, L. H. Stephens, and J. P. How, “Behavior classification algorithms at intersections and validation using naturalistic data,” in Proc. IEEE Intell. Vehicles Symp. (IV), Jun. 2011, pp. 601–606.

[15] P. Kumar, M. Perrollaz, S. Lefèvre, and C. Laugier, “Learning-based approach for online lane change intention prediction,” in Proc. IEEE Intell. Vehicles Symp. (IV), Jun. 2013, pp. 797–802.

[16] T. Streubel and K. H. Hoffmann, “Relational recurrent neural networks for vehicle trajectory prediction,” IEEE Trans. Intell. Transp. Syst., vol. 21, no. 4, pp. 129–140, Oct. 2019.

[17] L. Xu, J. Hu, H. Jiang, and W. Meng, “Establishing style-oriented driver models by imitating human driving behaviors,” IEEE Trans. Intell. Transp. Syst., vol. 16, no. 5, pp. 2522–2530, Oct. 2015.

[18] N. Deo, A. Rangesh, and M. M. Trivedi, “How would surround vehicles move? A unified framework for maneuver classification and motion prediction,” IEEE Trans. Intell. Vehicles, vol. 3, no. 2, pp. 129–140, Jun. 2018.

[19] N. Deo and M. M. Trivedi, “Convolutional social pooling for vehicle trajectory prediction,” in Proc. IEEE Intell. Transp. Syst. Conf. (ITSC), Oct. 2019, pp. 1813–1818.
[72] A. Paszke et al., “Pytorch: An imperative style, high-performance deep learning library,” in Proc. Adv. Neural Inf. Process. Syst., vol. 32, Dec. 2019, pp. 8026–8037.

[73] V. Alexiadis, J. Colyar, J. Halkias, R. Hranac, and G. McHale, “The next generation simulation program,” ITE J. Inst. Transp. Eng., vol. 74, no. 8, p. 22, Aug. 2004.

[74] R. Krajewski, J. Bock, L. Kloeker, and L. Eckstein, “The highD dataset: A drone dataset of naturalistic vehicle trajectories on German highways for validation of highly automated driving systems,” in Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC), Nov. 2018, pp. 2118–2125.

[75] W. Zhan et al., “INTERACTION dataset: An INTERnational, adversarial and cooperative meTION dataset in interactive driving scenarios with semantic maps,” 2019, arXiv:1910.03088.

[76] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014, arXiv:1412.6980.

[77] A. Kuefler, J. Morton, T. Wheeler, and M. Kochenderfer, “Imitating driver behavior with generative adversarial networks,” in Proc. IEEE Intell. Vehicles Symp. (IV), Jun. 2017, pp. 204–211.

[78] A. Kuefler, J. Morton, T. Wheeler, and M. Kochenderfer, “Imitating driver behavior with generative adversarial networks,” in Proc. IEEE Intell. Vehicles Symp. (IV), Jun. 2017, pp. 204–211.

[79] M. N. Fekri, A. M. Ghosh, and K. Grolinger, “Generating energy data for machine learning with recurrent generative adversarial networks,” Energies, vol. 13, no. 1, p. 130, Dec. 2019.

[80] L. Rosenblatt, X. Liu, S. Pouyanfar, E. D. Leon, A. Desai, and J. Allen, “Differentially private synthetic data: Applied evaluations and enhancements,” 2020, arXiv:2011.05537.

[81] I. Goodfellow, “NIPS 2016 tutorial: Generative adversarial networks,” 2017, arXiv:1701.00160.