

Self-Supervised Traffic Advisors: Distributed, Multi-view Traffic Prediction for Smart Cities

Jiankai Sun¹, Shreyas Kousik¹, David Fridovich-Keil², and Mac Schwager¹

Abstract—Connected and Autonomous Vehicles (CAVs) are becoming more widely deployed, but it is unclear how to best deploy smart infrastructure to maximize their capabilities. One key challenge is to ensure CAVs can reliably perceive other agents, especially occluded ones. A further challenge is the desire for smart infrastructure to be autonomous and readily scalable to wide-area deployments, similar to modern traffic lights. The present work proposes the Self-Supervised Traffic Advisor (SSTA), an infrastructure edge device concept that leverages self-supervised video prediction in concert with a communication and co-training framework to enable autonomously predicting traffic throughout a smart city. An SSTA is a statically-mounted camera that overlooks an intersection or area of complex traffic flow that predicts traffic flow as future video frames and learns to communicate with neighboring SSTAs to enable predicting traffic before it appears in the Field of View (FOV). The proposed framework aims at three goals: (1) inter-device communication to enable high-quality predictions, (2) scalability to an arbitrary number of devices, and (3) lifelong online learning to ensure adaptability to changing circumstances. Finally, an SSTA can broadcast its future predicted video frames directly as information for CAVs to run their own post-processing for the purpose of control.

I. INTRODUCTION

Connected and Autonomous Vehicles (CAVs) in smart cities have the potential to drastically improve road safety and traffic throughput. However, this potential is limited by the challenges of reliably perceiving other agents and predicting how they may move in complicated scenarios, such as unprotected left turns or pedestrian-dense areas. These challenges are compounded by varying lighting and weather, and by a lack of universal protocols for data sharing between CAVs and smart cities. To begin addressing these challenges, we propose a novel network of learning-enabled edge computers that broadcast universally accessible data about future occupied space at safety-critical locations such as busy intersections and hidden drives. We call this a Self-Supervised Traffic Advisor (SSTA), which is a statically mounted camera, paired with a small computer, overlooking a safety-critical location (see Fig. 1).

We envision an SSTA network providing advice to CAVs and enabling faster and safer traffic throughput in a fully distributed, decentralized, asynchronous way. On its own, an SSTA can learn to predict future camera measurements and, consequently, future occupied space within its field-of-view. By broadcasting time-varying predictions of occupancy to CAVs, SSTAs can advise future actions without requiring

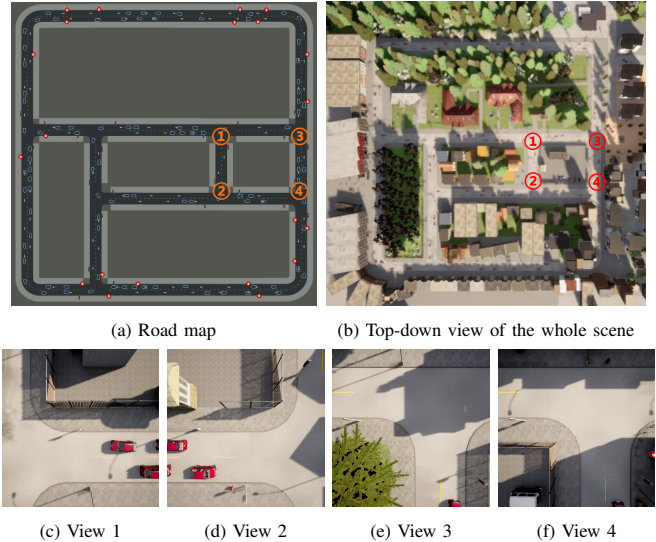


Fig. 1: **SSTA Overview**: Multiple intersections are connected according to the graph induced by a road network. At each intersection, a camera monitors the flow of traffic across the intersection, and the data is processed by an SSTA. Locations of each view are annotated in (a) and (b).

two-way communication. For example, if a CAV is waiting to make a left turn across traffic, but its view of oncoming traffic is occluded, it can passively receive an SSTA’s prediction to help decide when to begin its turn. By networking with each other, SSTAs learn to pass messages that improve each others’ predictive abilities. For example, a message received by an SSTA can inform it of a vehicle that is about to enter its field-of-view (FOV), resulting in a seamless prediction of traffic across devices and physical locations. Cameras already exist and are extensively deployed in modern transportation systems. To tackle the challenge of trustworthiness, existing defense and privacy-preserving techniques such as differential privacy (DP) can be applied in the future to make our approach robust and secure.

Contribution. The present work makes a first step towards realizing an SSTA network by proposing a decentralized, self-supervised traffic prediction architecture. We study our method’s abilities to learn inter-device communication, scale to many devices, and perform lifelong learning.

II. RELATED WORK

Our proposed method lies at the nexus of several applications of machine learning: video prediction, lifelong learning, and networked learning. While we note that there is extensive work on prediction, including for traffic occupancy, using

¹ Dept. of Aeronautics and Astronautics, Stanford University. ² Dept. of Aerospace Engineering and Engineering Mechanics, UT Austin. Corresponding author: jksun@stanford.edu. Toyota Research Institute provided funds to support this work.

supervised learning [1–3], here we focus on self-supervised learning from unlabeled, raw video.

Video Prediction. In modern video prediction, the objective is to estimate future frames of video given previous frames, typically using deep neural networks [4–6]. This task is attractive from a data collection perspective since it is often amenable to self-supervision. However, a key challenge is that individual frames are high-dimensional, resulting in a more difficult problem than just processing a single image; this has been tackled by using recurrent architectures [6, 7] (which we employ in the present work).

Video prediction typically focuses on a single camera, with the goal of either learning visual dynamics [5] or interpolating video spatially and temporally [8]. By contrast, the present work is concerned with *many* static, networked cameras, which resembles the surveillance and multi-view use cases in the literature. In surveillance, video prediction has been paired with background modeling to efficiently detect anomalies [9]. In the multi-view case, the challenge of a high-dimensional input is augmented, leading to a variety of efforts to interpolate or synthesize new views by leveraging information shared between views [10, 11]. In this work, we consider the multi-view case, but from far-apart cameras, and still seek to learn to share information.

Lifelong Learning. Also known as continual, sequential, or incremental learning, this is the process of learning from data that arrive sequentially, with only a small portion of input data available at a time [12]. The major challenge is that training a model on recent data may significantly impact its performance on past data; i.e., methods seek to “learn without forgetting” [13]. Most work in this space is concerned with supervised, task-based settings; while our setting is instead unsupervised, we are still concerned with saving the *best* data to train an SSTA, since storage space is limited.

There are a variety of approaches to leveraging past data. For example, one can store a subset of observed past data, and only take gradient steps from new data that do not increase the loss on the past data [14]. To avoid the memory and privacy concerns of saving past data, one can alternatively save encoded, low-dimensional representations [15, 16], or restrict gradient steps on new data to be perpendicular to the steps taken on past data [17]. Instead of constraining gradient steps, one can also focus on a model’s parameters; for example, one can keep learned parameters close to those trained for previous tasks [18], decide which parameters to update via an importance metric [19]. In this work, we test a variety of approaches to understand which is best for our problem setting.

Networked Learning. In the present work, we consider a framework that we call *networked co-learning*, where many agents train their own models simultaneously by passing messages and gradient information between one another. One can view this as a single, large, physically-distributed network; we leverage the local nature of each agent’s inputs and outputs to enable training the entire network in a distributed manner with only local information available for

Algorithm 1: SSTA

Input: Model parameter θ^i for node i , time step t , sequence length T , total number of nodes N
Output: Updated model parameter θ for all nodes

```

1 while not converged do
2   for  $t = 1, 2, \dots, T$  do
3     for  $i = 1, 2, \dots, N$  do
4       node  $i$  performs local prediction as (3)
         and broadcast  $y_t^i$  to other nodes
5     end for
6   end for
7   Update each node by minimizing (4)
8 end
9 return  $\theta$ 

```

each agent. We now distinguish this setting from two similar paradigms: distributed learning and graph neural networks.

In distributed and federated learning, one seeks to learn a *single* model that is shared among many agents [20]. In the distributed case, one seeks to use multiple distinct workers to train on centralized data; in the federated case, one uses decentralized data, with an emphasis on privacy by avoiding passing raw data on a network. There are a variety of architectures that seek a *consensus* of a centralized model in this setting [21, 22].

Graph neural networks (GNNs) model a function on data that are best represented by a graph [23]. This can be seen as a generalization of grid-like data structures (e.g., images), leading to insights such as convolutions or attention mechanisms over graph-structured data [24]. The key difference in the present work is that we consider a network of agents each training its own model (represented as a neural network); we do not seek to learn a model of the network (as a graph), but rather a network of models.

III. PROPOSED METHOD

We are interested in multi-view traffic prediction. Here, the key technical challenge is for each SSTA to determine which information is useful to communicate with its neighbors in order to improve their collective prediction performance. This communication is encoded in a learned latent space. We first define the *Networked Traffic Prediction* problem setup and how to address it using a latent space message-passing approach. We then discuss how we learn the recursive prediction task and leverage the conjugate relationships between each view. Finally, we present our algorithms for lifelong learning and continual deployment in the multi-view setting.

A. Networked Traffic Prediction

We consider a multi-view image prediction problem that arises in city traffic. As illustrated in Fig. 1, N intersections are connected according to the graph induced by a road network. At each intersection, a camera monitors the flow of traffic across the intersection, and the data is processed by an SSTA. More precisely, at each time t the i^{th} SSTA

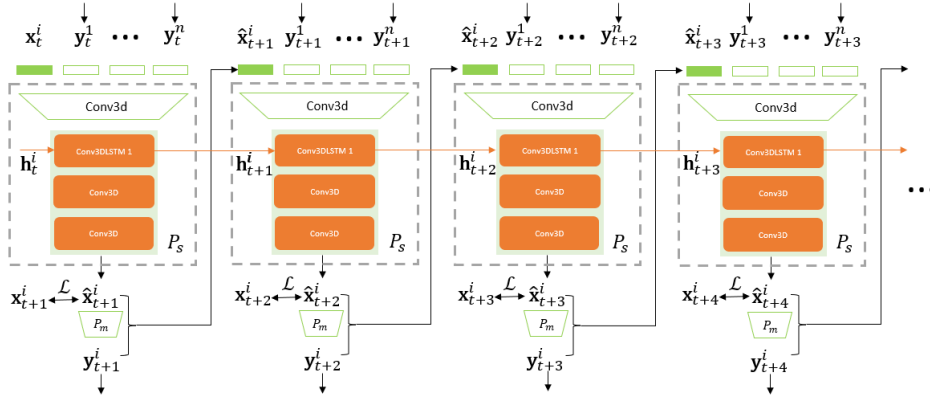


Fig. 2: The main architecture of a *single* SSTA i , in which the orange arrows denote the state transition paths over time. Note that messages $\{\mathbf{y}_s^j\}, j \neq i, s \geq t$ come from *other* SSTAs. \mathcal{L} is the loss between the prediction $\hat{\mathbf{x}}$ and ground-truth \mathbf{x} .

aims to match a sequence of estimated future images $\hat{\mathcal{X}}_t^i$ to the true images \mathcal{X}_t^i which are as yet unseen, where

$$\mathcal{X}_t^i = \{\mathbf{x}_\tau^i\}_{\tau=t}^{t+T} \text{ and } \hat{\mathcal{X}}_t^i = \{\hat{\mathbf{x}}_\tau^i\}_{\tau=t+1}^{t+T}, \quad (1)$$

with $\mathbf{x}_\tau^i \in \{0, 1, \dots, 255\}^{H \times W \times 3}$ denoting an RGB image of size $H \times W$, and $\hat{\mathbf{x}}_\tau^i$ a prediction of the same size. We denote the set of true image sequences over the entire network by $\mathbb{X}_t = \{\mathcal{X}_\tau^i\}_{i=1}^N$. That is, \mathbb{X}_t contains the image sequences for all SSTAs for all times $\tau \in \{t, \dots, t+T\}$.

We seek to optimize the image prediction quality at each SSTA. While many metrics for image quality exist [4], in this work we optimize the Mean Squared Error (MSE) for simplicity.

B. Message Passing

As discussed in Section II, video predictions are commonly expressed as functions of history produced by a parameterized model (for instance, $\hat{\mathbf{x}}_{t+1} = f(\hat{\mathbf{x}}_t; \theta)$ with parameters $\theta \in \mathbb{R}^p$). In this work, we introduce a novel functional architecture that enables each SSTA's prediction to depend upon past observations made by others. In particular, we design the i^{th} SSTA to output an encoded *message* \mathbf{y}_t^i as a real-valued vector of a user-specified dimension that is fixed *a priori*. Importantly, each SSTA sends a *single* message \mathbf{y}_t^i to some of its neighbors.

If the i^{th} SSTA is connected to a set of neighbors indexed by $K^i \subset \{1, \dots, N\}$, then, at time t , it receives a *message set*

$$\mathcal{Y}_t^i = \{\mathbf{y}_t^k\}_{k \in K^i}. \quad (2)$$

We denote the set of all messages received by all SSTAs at time t as $\mathbb{Y}_t = \{\mathcal{Y}_t^i\}_{i=1}^N$.

C. Recursive Prediction

The i^{th} SSTA must create a sequence of T predictions $\hat{\mathbf{x}}_t^i$ as images, which are a high-dimensional output. To reduce the output dimensionality of our model, we generate predictions recursively, one time step at a time. To enable this, the i^{th} SSTA maintains a hidden state \mathbf{h}_t^i (note, architecture implementation details are below in Section III-F). We initialize the hidden state as all zeros at time 0.

Then, each SSTA recursively generates predictions and messages while updating its hidden state:

$$(\hat{\mathbf{x}}_{\tau+1}^i, \mathbf{h}_{\tau+1}^i, \mathbf{y}_{\tau+1}^i) = \begin{cases} f(\mathbf{x}_t^i, \mathbf{h}_t^i, \mathcal{Y}_t^i; \theta^i) & \tau = t \\ f(\hat{\mathbf{x}}_\tau^i, \mathbf{h}_\tau^i, \mathcal{Y}_\tau^i; \theta^i) & \tau > t, \end{cases} \quad (3)$$

where θ^i denotes the trainable parameters of the i^{th} SSTA. Observe that this recursive model rollout depends on only the messages $\mathcal{Y}_t^i \in \mathbb{Y}_t$ available at time t ; during prediction ($\tau > t$) the SSTA discards all messages for times greater than t .

Also note that, when deployed (i.e., at test time), each SSTA predicts over a *receding time horizon*. That is, at time t it generates predictions $\hat{\mathbf{x}}_\tau^i$ according to (3) for $\tau \in \{t, \dots, t+T\}$. At time $t+1$, however, the process repeats; to initialize (3) in this instance we use the hidden state \mathbf{h}_{t+1}^i which was generated from the previous prediction rollout and discard that rollout's other hidden states from $\tau > 1$.

D. Networked Co-Learning

As discussed above, we seek to minimize the MSE between predicted and ground truth image sequences at all SSTAs simultaneously. The training loss associated to each time t in the training data is

$$\mathcal{L}_t(\mathbb{X}_t, \mathbb{Y}_t) = \sum_{i=1}^N \left(\sum_{\tau=t}^{t+T} \|\mathbf{x}_\tau^i - \hat{\mathbf{x}}_\tau^i\|_F^2 \right), \quad (4)$$

where ‘‘F’’ denotes the Frobenius norm. Unlike common federated or distributed learning frameworks [21, 25, 26], our goal is not for all of the SSTAs to learn a single, shared model, but rather for each to learn its own model, which we call *networked co-learning*.

To train the i^{th} SSTA, with gradient descent, we must compute the gradient of the overall loss with respect to the i^{th} set of parameters: $\nabla_{\theta^i} \mathcal{L}_t$. To do so efficiently, we observe that the i^{th} SSTA is only connected to the SSTAs in K^i . This sparse graphical dependence means that the aforementioned gradients may be computed in tandem with networks' message information, and each SSTA can therefore perform gradient descent independently on an individual loss \mathcal{L}_t^i given in the inner sum of (4).

To do so, the k^{th} SSTA, for $k \in K^i$, can compute its own *message loss* gradient $\nabla_{\mathbf{y}_t^i} \mathcal{L}_t^k$ and send it to the i^{th} SSTA in response to receiving the message \mathbf{y}_t^i . Given these partial derivatives, the i^{th} SSTA can compute

$$\nabla_{\theta^i} \mathcal{L}_t = \nabla_{\theta^i} \mathcal{L}_t^i + \sum_{k \in K^i} (\nabla_{\mathbf{y}_t^i} \mathbf{y}_t^i)^\top \nabla_{\mathbf{y}_t^i} \mathcal{L}_t^k, \quad (5)$$

via the chain rule. The complete networked co-training algorithm for our framework is shown in Alg. 1.

E. Lifelong Generative Learning and Continual Deployment

The advantage of using static cameras is that they can constantly train on data collected online. Unfortunately, it is not tractable to store all past data to reuse for training. A unique aspect of the SSTA framework is that we can train and deploy the system continually. We assume an infinite, continual stream of data. At each time step, the system receives a few consecutive samples $\{\mathbf{x}_t^i\}$ of recent traffic images. The goal is to continually learn and update the model that minimizes the prediction errors on previously seen and future samples. In other words, it aims at continuously updating and accumulating knowledge. Given an input model with parameters θ , the system at each time step reduces the empirical risk based on the recently received samples. The learning objective of the online system is Eq. (4).

F. Network Architecture

The SSTA associated with each camera view has an image prediction model P_s and a corresponding pre-trained message generator P_m , as Fig. 2 shows.

1) *Image Prediction Model P_s* : We use the Conv3DLSTM [27] architecture to model spatial and temporal information simultaneously. We build a network model by stacking several Conv3DLSTM layers, LayerNorm layer, and Conv3D layers to form an encoding-forecasting structure to predict next step state.

2) *Message Encoder Model P_m* : We build an auto-encoder (AE) with Conv3D layers and Conv3DTranspose layers. This AE model is pre-trained by reconstructing a portion of the images from the training set. We use the encoder of this pre-trained AE as the message generator at each time.

IV. EXPERIMENTS

Our experiments focus on the two core ideas underlying an SSTA network: 1) networked co-learning for video prediction, and 2) lifelong learning.

A. Datasets

We evaluate our method on *CARLA multi-view dataset* quantitatively and qualitatively. Due to the lack of a suitable multi-view traffic dataset, we collected data from the CARLA simulator [28–30]. Our *CARLA multi-view dataset* records the trajectories of consecutive multi-view traffic flows in default `Town02` in the form of top-down-view RGB video frames from cameras placed throughout the environment. The size of each top-down-view frame is $128 \times$

128×3 . The training set contains 11.2k frames from 8 camera locations while the validation set contains 4.8k frames from the same camera locations but from different times.

B. Metrics

We compare the baselines with the metrics such as Mean Square Error (MSE), Structural Similarity Index Measure (SSIM), Peak Signal to Noise Ratio (PSNR), and Learned Perceptual Image Patch Similarity (LPIPS) [31]. For PSNR and SSIM, a higher value denotes a better prediction performance. The value of SSIM ranges between -1 and 1, and a larger score means a greater similarity between two images. The difference between these metrics is that MSE estimates the absolute pixel-wise errors, PSNR/SSIM measures the similarity of structural information within spatial neighborhoods, while LPIPS is based on deep features and aligns more closely to human perceptions.

C. Implementation Details

We use the ADAM optimizer [32] to train the models with a mini-batch of 10 sequences. Unless otherwise specified, we set the learning rate to 10^{-3} and stop the training process after 100 epochs. We set the number of channels of each hidden state to 128 and the size of convolutional kernels inside the Conv3DLSTM unit to 5×5 . The quantitative results are averaged over 10 prediction timesteps.

D. Networked Co-Learning for Video Prediction

First, we evaluate the multi-view prediction capability of our model. In Fig. 3 and 4, we visualize some examples of the predicted results on the *CARLA multi-view dataset*, where the task is to predict the RGB images and the corresponding emerged messages for each subsequent step. From top to bottom, we see the ground truth, and prediction result for each view at each step.

Our model is able to predict a sequence of RGB images with corresponding emerged messages for multi-agent communication. Specifically, the most challenging part in multi-view traffic prediction is the bridging of traffic flow “out” and “in” in different views, where communication with messages plays an important role, and can only be inferred from context and sequence relationships between multiple views.

We further show our model’s performance as the number of views increases. Intuitively, the higher the number of views, the more complex the task, as it involves more messages to communicate. From the results of Table I and Fig. 3 and 4, our framework can effectively predict future traffic not only when the number of views is small (2-view), but also when the number of views is large (4/8-view). This scaling is made possible by SSTAs’ ability to learn compact message formats which include only the information relevant for others to improve their predictions.

E. Lifelong Learning

In a practical intelligent transportation application scenario, it will be critical for SSTAs to learn continuously based

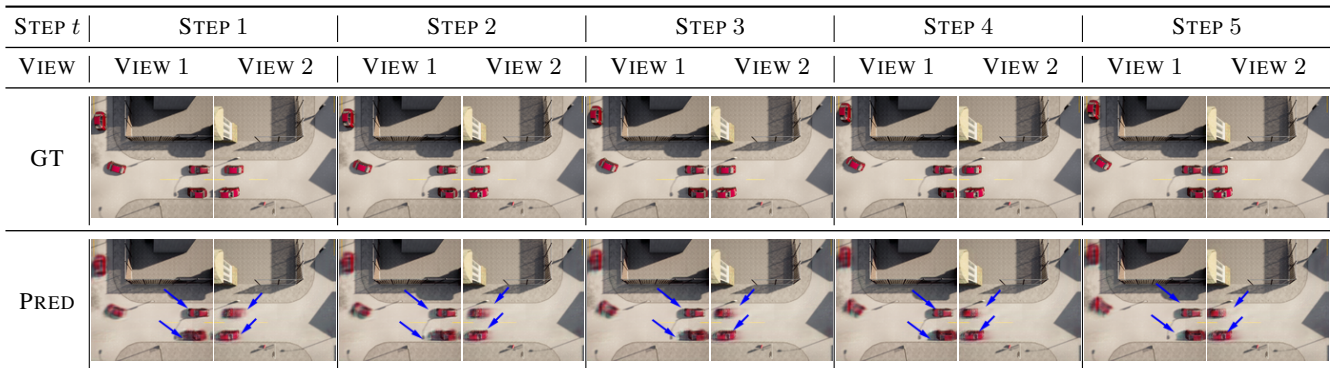


Fig. 3: **Qualitative Results:** 2-view prediction. We highlight the cars in the views related to the messages with blue arrows to show the message utility.

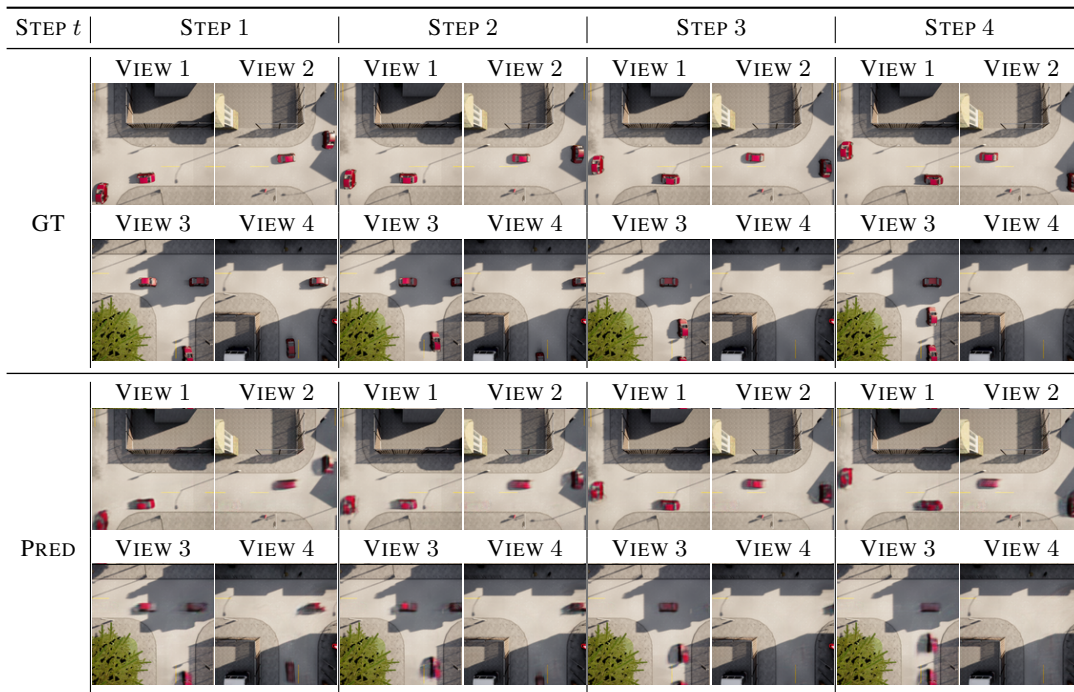


Fig. 4: **Qualitative Results:** 4-view prediction.

# Views	MSE↓	SSIM↑	PSNR↑	LPIPS↓
2-view	0.16	0.9661	29.64	0.0513
4-view	0.16	0.9625	29.30	0.0535
8-view	0.18	0.9501	28.41	0.0581

TABLE I: **Scalability:** Performance of our framework SSTA on different number of views.

Method	MSE↓	SSIM↑	PSNR↑	LPIPS↓
SW ($D = 50$)	0.24	0.8912	23.39	0.1199
SW ($D = 150$)	0.22	0.9061	24.59	0.1001
SW ($D = 300$)	0.21	0.9175	26.18	0.0801
ID ($D = 300$)	0.19	0.9481	28.35	0.0603

TABLE II: **Lifelong Learning:** influence of different strategies and different sizes of the sliding window D . SW: Sliding Window, ID: Interesting Data.

on streaming input data, and update the model parameters in real-time. Hence, we seek to understand the effects of different strategies for lifelong learning on our video prediction task. We assume an SSTA can store up to $D \in \mathbb{N}$ time steps worth of data, and compare the following strategies:

- 1) *Sliding Window (SW)*. At each time t , each SSTA continually trains on the past D time steps of data.
- 2) *Interesting Data (ID)*. The ID strategy is modified based on the task-free continual learning framework [33]. At

each time t , we save the latest datum for future training if the norm of the gradient $\|\nabla_{\theta^i} \mathcal{L}_t^i\|_2$ is larger than the average norm of past data gradients.

We also ablate over different sizes of the sliding window D . From the results in Table II, we can see that increasing the size of the sliding window D and using the ID training strategy is useful. Quantitative results validate the effectiveness of our framework in the lifelong learning setting and demonstrate its potential for deployment in traffic scenarios.

V. CONCLUSION

We present Self-Supervised Traffic Advisors, a learning framework that leverages self-supervised video prediction to autonomously predict traffic throughout a smart city. We argue that communication and co-training are useful for this multi-view prediction. Our experimental results show that the SSTA framework achieves three goals: (1) inter-device communication to enable high-quality predictions, (2) scalability to an arbitrary number of devices, and (3) lifelong online learning to ensure adaptability to changing circumstances. Our work constitutes a major step towards the goal of enabling multiple autonomous agents to predict real-world streaming data. Future work will be devoted to improving prediction quality, especially in the transition region in different views, and integrating SSTAs with downstream tasks such as vehicle motion planning in complex traffic scenarios. Our framework can also be combined with privacy-preserving techniques, such as differential privacy.

REFERENCES

- [1] Masha Itkina, Katherine Driggs-Campbell, and Mykel J Kochenderfer. “Dynamic environment prediction in urban scenes using recurrent representation learning”. In: *ITSC*. IEEE. 2019, pp. 2052–2059.
- [2] Yunbo Wang, Mingsheng Long, Jianmin Wang, Zhifeng Gao, and Philip S Yu. “PredRNN: Recurrent Neural Networks for Predictive Learning using Spatiotemporal LSTMs”. In: *NeurIPS*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017.
- [3] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. “Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data”. In: *preprint arXiv:2001.03093* (2020).
- [4] Michael Mathieu, Camille Couprie, and Yann LeCun. “Deep multi-scale video prediction beyond mean square error”. In: *preprint arXiv:1511.05440* (2015).
- [5] Jacob Walker, Carl Doersch, Abhinav Gupta, and Martial Hebert. “An uncertain future: Forecasting from static images using variational autoencoders”. In: *ECCV*. Springer. 2016, pp. 835–851.
- [6] Mohammad Babaeizadeh, Chelsea Finn, Dumitru Erhan, Roy H. Campbell, and Sergey Levine. “Stochastic Variational Video Prediction”. In: *ICLR*. 2018.
- [7] Jingwei Xu, Bingbing Ni, Zefan Li, Shuo Cheng, and Xiokang Yang. “Structure preserving video prediction”. In: *CVPR*. 2018, pp. 1460–1469.
- [8] Chaochao Lu, Michael Hirsch, and Bernhard Scholkopf. “Flexible spatio-temporal networks for video prediction”. In: *CVPR*. 2017, pp. 6523–6531.
- [9] Xianguo Zhang, Tiejun Huang, Yonghong Tian, and Wen Gao. “Background-modeling-based adaptive prediction for surveillance video coding”. In: *TIP* 23.2 (2013), pp. 769–784.
- [10] Shruti Vyas, Yogesh S Rawat, and Mubarak Shah. “Multi-view action recognition using cross-view video prediction”. In: *ECCV*. Springer. 2020, pp. 427–444.
- [11] Bowen Pan, Jiankai Sun, Ho Yin Tiga Leung, Alex Andonian, and Bolei Zhou. “Cross-view semantic segmentation for sensing surroundings”. In: *RA-L* 5.3 (2020), pp. 4867–4873.
- [12] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. “Continual lifelong learning with neural networks: A review”. In: *Neural Networks* 113 (2019), pp. 54–71.
- [13] Matthias Delange et al. “A continual learning survey: Defying forgetting in classification tasks”. In: *TPAMI* (2021).
- [14] David Lopez-Paz and Marc’Aurelio Ranzato. “Gradient episodic memory for continual learning”. In: *NeurIPS* 30 (2017).
- [15] Zhizhong Li and Derek Hoiem. “Learning without forgetting”. In: *TPAMI* 40.12 (2017), pp. 2935–2947.
- [16] Amal Rannen, Rahaf Aljundi, Matthew B. Blaschko, and Tinne Tuytelaars. “Encoder Based Lifelong Learning”. In: *ICCV*. Oct. 2017.
- [17] Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. “Orthogonal gradient descent for continual learning”. In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2020, pp. 3762–3773.
- [18] Sang-Woo Lee, Jin-Hwa Kim, Jaehyun Jun, Jung-Woo Ha, and Byoung-Tak Zhang. “Overcoming catastrophic forgetting by incremental moment matching”. In: *NeurIPS* 30 (2017).
- [19] Friedemann Zenke, Ben Poole, and Surya Ganguli. “Continual learning through synaptic intelligence”. In: *ICML*. PMLR. 2017, pp. 3987–3995.
- [20] Chaoyang He et al. “Fedml: A research library and benchmark for federated machine learning”. In: *preprint arXiv:2007.13518* (2020).
- [21] Javier Yu, Joseph Vincent, and Mac Schwager. “DiNNO: Distributed Neural Network Optimization for Multi-Robot Collaborative Learning”. In: *RA-L* (2022).
- [22] Ilai Bistriz, Ariana Mann, and Nicholas Bambos. “Distributed distillation for on-device learning”. In: *NeurIPS* 33 (2020), pp. 22593–22604.
- [23] Jie Zhou et al. “Graph neural networks: A review of methods and applications”. In: *AI Open* 1 (2020), pp. 57–81.
- [24] Jiachen Li, Hengbo Ma, Zhihao Zhang, Jinning Li, and Masayoshi Tomizuka. “Spatio-Temporal Graph Dual-Attention Network for Multi-Agent Prediction and Tracking”. In: *TITS* (2021), pp. 1–14.
- [25] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. “Federated learning: Challenges, methods, and future directions”. In: *IEEE Signal Processing Magazine* 37.3 (2020), pp. 50–60.
- [26] Peter Kairouz et al. “Advances and open problems in federated learning”. In: *preprint arXiv:1912.04977* (2019).
- [27] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. “Convolutional LSTM network: A machine learning approach for precipitation nowcasting”. In: *NeurIPS* 28 (2015).
- [28] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. “CARLA: An Open Urban Driving Simulator”. In: *CoRL*. 2017, pp. 1–16.
- [29] Junning Huang et al. “Learning a Decision Module by Imitating Driver’s Control Behaviors”. In: *CoRL*. Vol. 155. Proceedings of Machine Learning Research. PMLR, 16–18 Nov 2021, pp. 1–10.
- [30] Jiankai Sun, Hao Sun, Tian Han, and Bolei Zhou. “Neuro-Symbolic Program Search for Autonomous Driving Decision Module Design”. In: *CoRL*. Vol. 155. Proceedings of Machine Learning Research. PMLR, 16–18 Nov 2021, pp. 21–30.
- [31] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. “The Unreasonable Effectiveness of Deep Features as a Perceptual Metric”. In: *CVPR*. Los Alamitos, CA, USA: IEEE Computer Society, June 2018, pp. 586–595.
- [32] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *ICLR*. Ed. by Yoshua Bengio and Yann LeCun. 2015.
- [33] Rahaf Aljundi, Klaas Kelchermans, and Tinne Tuytelaars. “Task-free continual learning”. In: *CVPR*. 2019, pp. 11254–11263.