2024 IEEE 27th International Conference on
Intelligent Transportation Systems (ITSC)
September 24- 27, 2024. Edmonton, Canada

Comparative Studies of Neural Operators For Traffic State Estimation

Qi Gao, and Xuan Di* *Member, IEEE*

***Abstract*— Traffic State Estimation (TSE) is essential for traffic management, planning, and control by predicting future traffic conditions such as flow density and average velocity. Traditional TSE methods, which employ either numerical techniques or deep learning, often require extensive prior knowledge and are too time-consuming for real-time applications. Recently, Neural Operators (NOs), a novel type of machine learning model, have been recognized for their efficiency in learning complex mappings between functional spaces. NOs offer significant advantages, including the elimination of the need for recalculations or retraining under varying initial or boundary conditions, and they are trained directly with data, without understanding of the underlying physical laws. This capability makes NOs particularly suitable for real-world, time-sensitive applications. Although NOs have been successfully used to model a range of phenomena from simple to complex partial differential equations, their ability to generalize across diverse functional spaces in traffic flow has not been thoroughly explored. This paper uses simulated traffic flow data with varying initial conditions and real-world traffic flow data from the Next Generation SIMulation (NGSIM) dataset to evaluate the capacity of different NOs to approximate various traffic flow scenarios. Our study aims to guide transportation professionals in selecting suitable NOs and to encourage further advancements in NO research.**

I. Introduction

In the realm of traffic management, the ability to accurately predict traffic flow states, such as density and velocity, is crucial for effective control and planning. Traditional methods for Traffic State Estimation (TSE) have largely relied on numerical techniques[1] or physics informed deep learning approaches[2], [3]. While these methods have been instrumental in understanding traffic dynamics, they typically require extensive computation, making them impractical for real-time applications. This has led to an interest in more efficient methodologies capable of quick adaptation to changing conditions.

Recent advances in machine learning have introduced Neural Operators (NOs)[4], [5], [6], [7], [8], a promising class of models adept at learning complex mappings between function spaces directly from data. Unlike traditional models, NOs obviate the need for repeated recalculations under new conditions, offering a significant leap in operational efficiency, which is especially useful in the field of transportation [9], [10]. This paper explores the application of various Neural Operators predict traffic flow, using both simulated and real-world data from the Next Generation Simulation (NGSIM) dataset. Our study is designed to evaluate the adaptability of NOs to diverse traffic scenarios, assessing their capabilities across different functions commonly seen in traffic flow.

By comparing several types of NOs, this research aims to provide insights into selecting the most appropriate models for specific traffic conditions and to foster further developments in the field of traffic state estimation using Neural Operators. In the process of doing so, we also discovered some insights in NOs, which we believe will also be instrumental to improvement of existing NOs and developments of future NOs.

Contributions

The main contribution of the paper can be summarized as follow:

1) Demonstrated effective methods for adapting the existing architecture of NOs to handle multi-channel inputs and outputs.
2) Proposed a procedure for preparing complex boundary conditions for TSE, making them suitable for integration with the FNO family of NOs.
3) Proposed a set of simulated traffic flow datasets, created using various traffic flow models and different initial conditions to enhance the evaluation of NO performance.
4) Performed an in-depth evaluation of various NOs across two datasets, comprehensively representing diverse TSE scenarios.

The rest of the paper is organized as follow: Sec. II introduces two commonly used macroscopic traffic flow model. Sec. III introduces 5 different existing NOs. Sec. IV discuss how we conducted our experiment for the comparison, including how the two datasets are generated and how NOs are trained with these datasets. In the next Sec. V the main results of this paper are presented followed by the Sec. VI concludes this study.

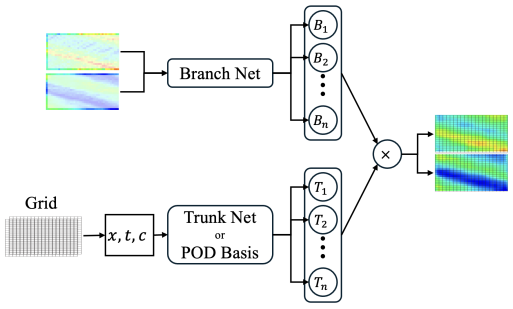

Fig. 1: Generalized architecture for DeepONet and POD-DeepONet.

*Corresponding author: Xuan Di.

‡This work is sponsored by NSF CPS-2038984.

Qi Gao is with the Department of Civil Engineering and Engineering Mechanics, Columbia University, New York, NY, 10027, USA (E-mail: qg2179@columbia.edu).

Xuan Di is with the Department of Civil Engineering and Engineering Mechanics, Columbia University, New York, NY, 10027 USA, and also with the Data Science Institute, Columbia University, New York, NY, 10027 USA (E-MAIL: sharon.di@columbia.edu).

Code avaliable at https://github.com/CU-DitecT/TSE_NOs_Compare.git

2024 IEEE 27th International Conference on Intelligent Transportation Systems (ITSC) | 979-8-3315-0592-9/24/$31.00 ©2024 IEEE | DOI: 10.1109/ITSC58415.2024.10920089

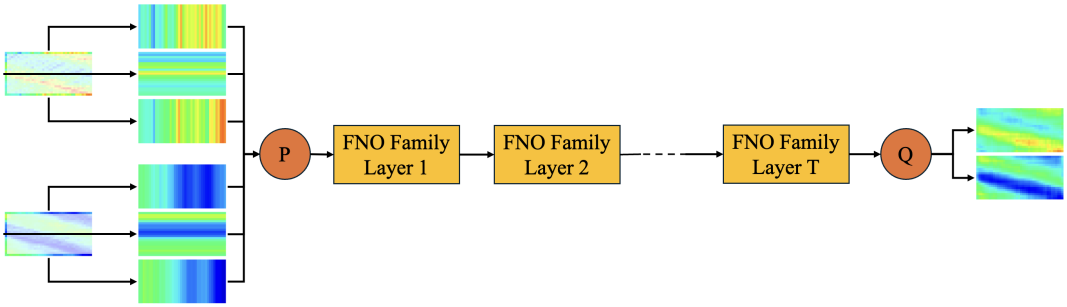

Fig. 2: Generalized architecture for FNO family in handling multi channel inputs and outputs. The whole FNO family shares this pipeline, including TFNO and MWT, but with very different design in their layers. Though they do all use Spectral Convolutional Layer as a core component.

II. Macroscopic traffic flow models

Macroscopic traffic flow models are a class of mathematical models used to describe and analyze the flow of traffic as a whole, rather than focusing on individual vehicles. These models treat traffic flow similarly to fluids, using equations and concepts from fluid dynamics to capture the overall behavior of traffic on road networks. The main components of macroscopic traffic models include flow f (the number of vehicles passing a point per unit of time), density ρ (the number of vehicles per unit length), and speed u (the distance covered per unit time). Commonly used macroscopic traffic models include but are not limited to Lighthill-Whitham-Richards (LWR) [11], [12] traffic model, Cell Transmission Models (CTM) [13] and Aw, Rascle[14] and Zhang[15] (ARZ) traffic model. In this paper, we focus mainly on LWR and ARZ since they are hyperbolic PDEs, where one major objective of NOs is to learn the mapping between initial and results of PDEs.

a) LWR Model: The Lighthill-Whitham-Richards (LWR) model is a continuum traffic flow model, Developed independently by Sir Michael James Lighthill and Gerald Beresford Whitham in 1955, and extended by Peter Richards in 1956, that describes how the density of cars on a road affects the speed at which they travel. This relation is called fundamental diagram, of which, the most common version is characterized by $u(x,t) = u_{\text{free}}(1-\rho(x,t)/\rho_{\text{jam}})$. Where u_{free} stands for the maximum speed a vehicle can reach when the high way is empty, and ρ_{jam} stands for the maximum density of vehicle on the high way (i.e. maximum number of vehicle on unit length of highway). The flux function f is intuitively defined as the product of speed u and density ρ as $f(x,t) = \rho(x,t)u(x,t)$. Therefore, the hyperbolic conservation law can be characterized by the following PDE:

$$\partial_t \rho(x,t) + \partial_x f(x,t) = 0 \tag{1}$$

b) ARZ model: The Aw, Rascle[14] and Zhang [15] (ARZ) model is a second-order traffic flow model developed independently by Aw, Rascle and Zhang to address an unrealistic assumption in the first-order LWR model. The LWR model assumed that vehicles, having no momentum, can instantaneously adjust their speed based on the current traffic density—an assumption implying that vehicles either have zero mass or possess infinite power. By contrast, the ARZ model incorporates vehicle momentum into its formulation, offering a more realistic depiction of traffic dynamics, which is formulate as follow:

$$\partial_t \rho + \partial_x \rho u = 0 \tag{2}$$

$$\partial_t(\rho\,(u + p(\rho))) + \partial_x(\rho\,(u + p(\rho))\,u) = 0 \tag{3}$$

Where ρ is the traffic density and u is the speed, which is the same as LWR. $p(\rho)$ is the pressure function, an increasing function. In this paper, we focus on pressure function of form $p(\rho) = \rho^{\gamma}$, where $\gamma = 1$. For solving the equation numerically in the experiment section, let $z = \rho\,(u + p(\rho))$, then we have another form of eq(4) $\partial_t z + \partial_x z u = 0$.

III. Neural Operators

In this section, we examine two widely used Neural Operators (NOs), Fourier Neural Operator(FNO) [6] and Deep Operator Networks(DeepONet) [4], along with their recent variants, Multiwavelet-based Model(MWT) [8], Tensorized Fourier Neural Operator (TFNO) [7] and POD-DeepONet[5]. These NOs aim to learn a mapping $v \to u$ between two function spaces $\mathscr{V}$ and $\mathscr{U}$, employing different methodologies. This mapping often represents the relationship between the initial conditions and result of a specific Partial Differential Equation (PDE).

a) DeepONet: Inspired by previous work on approximating non-linear operator by neural networks [16], Deep Operator Networks(DeepONet) [4] stand as pioneering of NOs. DeepONet featured two neural networks (two group of neural networks in variants), known as the branch net b and the trunk net t. The trunk net functions as a basis set, taking position ξ as input and produce $t(\xi)$, which represents the values of each basis function at ξ. On the contrary, the branch net processes the discredited input function, generating $b(v)$ as the coefficients for each basis to approximate the output function u. Consequently, the output u in ξ is estimated by $u(\xi) = b(v)^T t(\xi)$

b) POD-DeepONet: POD-DeepONet[5], a variant of DeepONet, differs mainly in its use of a predetermined basis to replace the trunk net, while maintaining the branch net as a neural network learned through training. This basis is established through Principal Component Analysis (PCA) on the test set prior to training. This approach significantly reduced training time and improved performance.

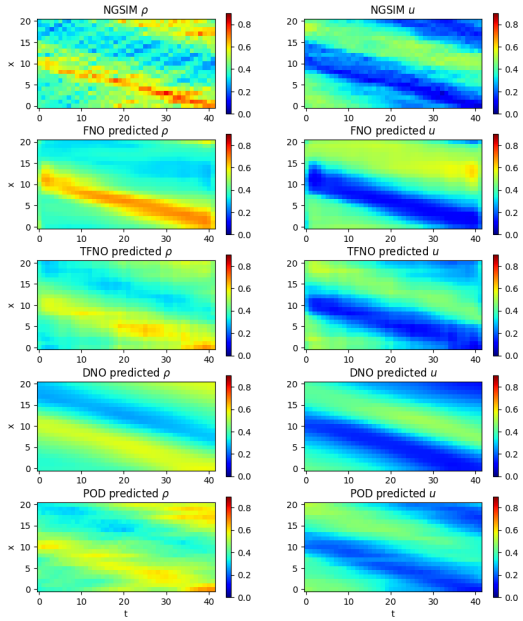

Fig. 3: Visualization of each NOs' prediction given the same initial and boundary condition from NGSIM data.

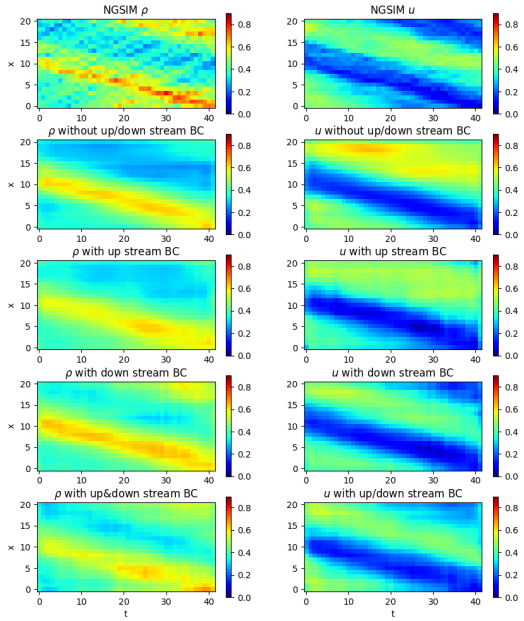

Fig. 4: Visualization of TFNO's prediction given the different initial and boundary condition

c) FNO: The Fourier Neural Operator (FNO) [6] has emerged as a popular Neural Operator due to its unique approach and general applicability. Unlike DeepONet, which learns basis functions from data, the FNO utilizes sine and cosine functions as the basis. These functions are well studied in their ability to approximate any function within a Hilbert space, given a sufficiently high mode. This choice allows the basic building block of FNO, Spectral Convolution layer, to leverage the existing PyTorch Fast Fourier Transform package that allow flow of gradients through both forward and backward Fourier transforms. Consequently, the FNO efficiently facilitates the transfer of information between the functional and coefficient spaces across each layer of the network. In addition, the FNO incorporates innovative features such as high-frequency filtering and skip connections, drawing inspiration from the ResNet architecture. These enhancements contribute to its robustness and versatility to learn complex mapping between initial functions and terminal function.

d) MWT: As a derivative of the Fourier Neural Operator (FNO), the Multiwavelet-Based Neural Operator (MWT) [8] retains many aspects of the FNO's design including the general archacture and the Spectral Convolution layer. However, it make significant change to the original FNO layers. Instead of directly feed the input of a layer directly in to the Spectral Convolution layer and skipping layer, MWT continuously conduct multiwavelet transform until reaching the smallest wavelet of the input. Then MWT will feed the output of multiwavelet transform in to three different Spectral Convolution layer separately and reassamble the output through inverse multiwavelet transform. This adjustment enables the MWT to capture and learn complex high-frequency information more effectively than the FNO.

As a derivative of the Fourier Neural Operator (FNO), the Multiwavelet-Based Neural Operator (MWT)[8], retains many features of the FNO's design, including the general architecture and the Spectral Convolution layer. However, it introduces significant modifications to the original FNO layers. Unlike the FNO, which feeds the input of a layer directly into the Spectral Convolution layer and includes a skip connection, the MWT performs successive multiwavelet transformations until it reaches the smallest wavelet scale of the input. Following this, the MWT processes the output from the multiwavelet transform through three distinct Spectral Convolution layers separately. The outputs from these Spectral Convolution layers are then reassembled via an inverse multiwavelet transform as output of a MWT layer. This modification allows the MWT to more effectively capture and learn complex high-frequency information compared to the FNO.

e) TFNO: The Tensorized Fourier Neural Operator (TFNO) [7] is a recent variant of FNO that aims to improve efficiency through Tucker decomposition. While improving efficiency, TFNO also introduced some improvement to FNO including normalization, channel mixing and soft-gated skip-connection. These modifications to FNO significantly improved performance and therefore should be considered separately from the original FNO.

IV. Experiment

A. NGSIM Dataset

The NGSIM dataset [17], provided by the US Federal Highway Administration, contains detailed vehicle trajectory data for multiple highways and urban streets. Our study focuses on a segment of US Highway 101, monitored through a camera on a tall building on June 15, 2005. The dataset

Inputs	FNO	TFNO	DeepONet	POD-DeepONet
ρ	0.2355	0.2038	0.2269	0.2163
ρ w/ upstream	0.2215	0.2023	0.2288	0.2006
ρ w/ downstream	0.2194	0.1650	0.1890	0.1775
ρ w/ up&downstream	0.2158	0.1576	0.1772	0.1734

TABLE I: L2 Error of NOs predicting NGSIM data

covers vehicle speeds and locations over approximately 680 meters for 2,770 seconds.

The derivation of traffic density ρ and speed u follows established methods [18], [19]. Initially, vehicles not consistently visible in the video are filtered out. The highway is segmented into 21 spatial cells, and the observation period is segmented into 1,770 temporal cells. For each cell, we calculate the number of vehicles and their average speed across all lanes to determine the traffic density and speed.

To prevent the neural operators (NOs) from merely learning to map similar initial conditions to corresponding outcomes, we first divide all 1,770 temporal cells into 30 subsections of 59 cells each with no overlapping. Then we randomly allocate 25 subsections to the training set and 5 to the testing set. In this way, we can ensure the NOs capture the underlying traffic dynamics, allowing us to compare their performance in learning these dynamics from data.

For effective neural operator (NO) training, we require a large number of initial and boundary condition pairs. In this work, we chose to divide the data into windows spanning 21 spatial cells by 42 temporal cells—sufficiently large to observe shockwave effects in traffic flow but small enough to ensure an adequate number of training pairs. These windows are generated by applying sliding windows to each subsection from the training and testing sets. We then randomly shuffle the windows within the training set and the testing set separately. Eventually, we end up with a training size of 450 and a testing size of 90. While this dataset seems small, the NOs discussed here are usually compared on a dataset containing 1,000 training pairs and 200 testing pairs provided by the authors of the Fourier Neural Operator (FNO) [6]. Our dataset is smaller but still of similar magnitude. Therefore, we believe this dataset can adequately reflect the performance of NOs.

B. Numerical Simulated Traffic Flow Data

1) Numerical Schemes: The numerical simulated traffic flow data used for training NOs are generated by solving LWR and ARZ with Lax-Friedrichs scheme. The Lax-Friedrichs scheme is a finite difference method method used to solve partial differential equations, particularly hyperbolic conservation laws, which are often found in physics and engineering applications such as fluid dynamics and traffic flow. This scheme is particularly popular for its simplicity and robustness. Therefore, we use Lax-Friedrichs scheme to solve the LWR model and the ARZ model for generating training data for NOs.

The Lax-Friedrichs scheme for solving LWR is as follow:

$$\rho(x,t+\Delta t) = 0.5\cdot(\rho(x-\Delta x)+\rho(x+\Delta x)) \tag{4}$$
$$-0.5\frac{\Delta t}{\Delta x}(\rho(x+\Delta x,t)\cdot u(x+\Delta x,t) \tag{5}$$
$$-\rho(x-\Delta x,t)\cdot u(x-\Delta x,t)) \tag{6}$$
$$u(x,t+\Delta t) = u_{\text{free}} * (1-\frac{\rho(x,t+\Delta t)}{\rho_{\text{jam}}}) \tag{7}$$

The Lax-Friedrichs scheme for solving ARZ is as follow:

$$\rho(x,t+1) = 0.5\cdot(\rho(x-\Delta x)+\rho(x+\Delta x)) \tag{8}$$
$$-0.5\frac{\Delta t}{\Delta x}(\rho(x+\Delta x,t)\cdot u(x+\Delta x,t) \tag{9}$$
$$-\rho(x-\Delta x,t)\cdot u(x-\Delta x,t)) \tag{10}$$
$$z(x,t+1) = 0.5\cdot(z(x-\Delta x)+z(x+\Delta x)) \tag{11}$$
$$-0.5\frac{\Delta t}{\Delta x}(z(x+\Delta x,t)\cdot u(x+\Delta x,t) \tag{12}$$
$$-z(x-\Delta x,t)\cdot u(x-\Delta x,t)) \tag{13}$$
$$u(x,t+1) = \frac{z(x,t+1)}{\rho(x,t+1)} - \rho^{\gamma}(x,t+1) \tag{14}$$

For the simulation, we focus on periodic boundary condition with in domain $x \in [0,1]$, i.e. $\rho(0,t) = \rho(1,t)$ and $u(0,t) = u(1,t)$. Then given the initial condition $\rho_0(x)$ and $u_0(x)$ (which will be discussed in following section), we use the above scheme to solve for $\rho(x,t)$ and $z(x,t)$ for $x \in [0,1]$ and $t \in [0,1]$.

There is a diffusive effect in solution of Lax-Fridrichs scheme due to it's averaging of neighboring points. This could lead to larger error in the results when there is sharp gradient or discontinuities. Fortunately, this diffusive effect depends on the size of grid, because it only average neighbor points. So we also solve PDEs in finer grid ($\Delta x = \Delta t = 2^{-13}$) and down sample to the grid ($\Delta x = \Delta t = 2^{-7}$) of the training dataset. The solution in the grid of training dataset is also preserved, given the dataset used in previous works usually contain diffusion terms.

2) Initial Condition Generation: In simulating traffic flow, the function form of the result largely depends on the initial boundary conditions. Therefore, to create a comprehensive traffic flow dataset encompassing all types of functions, a diverse range of initial boundary conditions is essential. Below, we detail the types of initial boundary conditions employed, their corresponding functional properties, and our methods for generating them.

Fourier Series This initial condition set comprises functions represented as sums of sinusoidal and cosinusoidal components, with coefficients randomly sampled for each. The series considers specific frequencies up to modes 2 or 4. These functions are ideal for representing smooth behaviors effectively.

Sinusoidal Waves w/ Phase Shift(PS) This set includes sinusoidal wave functions with frequencies randomly sampled up to 8, each incorporating a random phase shift. This configuration is designed to evaluate the ability of numerical operators (NOs) to handle high-frequency functions accurately.

Bell Shape Characterized by a smooth, single-peaked function that closely resembles a Gaussian distribution, the coefficients for the peak's position and the standard deviation are randomly determined. This initial condition is used to assess each NO's capability in managing sharp transitions or shocks within the function.

Piece-Wise Constant This initial condition consists of functions defined by three regions of constant values, each separated by abrupt changes. The values and length for each region are randomly chosen. It is particularly useful for

Dataset	TFNO	FNO	MWT	DeepONet	DeepONet(POD)
Fourier Series(mode=2, gird=2^7)	0.00255	0.01355(5.33)	0.00145(0.57)	0.01436(5.64)	0.00670(2.63)
Fourier Series(mode=4, gird=2^7)	0.00306	0.01524(5.16)	0.00214(0.70)	0.01661(5.43)	0.00840(2.75)
Sin Waves w/ PS (gird=2^7)	0.00239	0.01345(5.63)	0.00179(0.75)	0.01372(5.75)	0.00610(2.56)
Bell Shape(gird=2^7)	0.00356	0.01773(4.64)	0.00225(0.63)	0.01886(5.29)	0.00733(2.06)
Mixed(grid=2^7)	0.00135	0.01637(12.14)	0.00106(0.79)	0.01434(10.64)	0.00548(4.06)
Fourier Series(mode=2, gird=2^{13})	0.00761	0.03081(4.05)	0.00907(1.19)	0.05672(7.46)	0.04929(6.48)
Fourier Series(mode=4, gird=2^{13})	0.01195	0.03326(2.78)	0.01287(1.08)	0.06566(5.49)	0.07210(6.03)
Sin Waves w/ PS (gird=2^{13})	0.00649	0.03312(5.11)	0.00999(1.54)	0.06437(9.93)	0.06453(9.95)
Bell Shape(gird=2^{13})	0.00892	0.03746(4.20)	0.00975(1.09)	0.06652(7.46)	0.04643(5.21)
Piece-wise Constant(grid=2^{13})	0.02496	0.03900(1.56)	0.03338(1.34)	0.12623(5.06)	0.12401(4.97)
Mixed(grid=2^{13})	0.00698	0.03759(5.38)	0.00708(1.01)	0.05879(8.42)	0.03981(5.70)

TABLE II: L2 error of NOs in predicting numerically simulated results of LWR equation with different initial conditions

Dataset	TFNO	FNO	MWT	DeepONet	DeepONet(POD)
Fourier Series(mode=2, gird=2^7)	0.00857	0.02247(2.62)	0.00676(0.79)	0.05611(6.54)	0.03235(3.77)
Fourier Series(mode=4, gird=2^7)	0.00949	0.02783(2.93)	0.00880(0.93)	0.07102(7.49)	0.04492(4.73)
Sin Waves w/ PS (gird=2^7)	0.01068	0.02701(2.53)	0.00972(0.91)	0.08313(7.79)	0.05536(5.18)
Bell Shape(gird=2^7)	0.01016	0.03101(3.05)	0.00798(0.79)	0.06070(5.98)	0.03677(3.62)
Mixed(grid=2^7)	0.00454	0.02767(6.10)	0.00415(0.92)	0.04205(9.27)	0.01720(3.79)
Fourier Series(mode=2, gird=2^{13})	0.01716	0.04072(2.37)	0.02072(1.21)	0.11490(6.69)	0.08560(4.99)
Fourier Series(mode=4, gird=2^{13})	0.02644	0.04934(1.87)	0.03026(1.14)	0.16212(6.13)	0.12120(4.58)
Sin Waves w/ PS (gird=2^{13})	0.02890	0.05206(1.80)	0.03451(1.19)	0.17872(6.18)	0.15728(5.44)
Bell Shape(gird=2^{13})	0.02517	0.05434(2.16)	0.02644(1.05)	0.13382(5.32)	0.09674(3.84)
Piece-wise Constant(grid=2^{13})	0.06626	0.06497(0.98)	0.07234(1.09)	0.24581(3.71)	0.17707(2.67)
Mixed(grid=2^{13})	0.02076	0.05623(2.71)	0.02326(1.12)	0.11130(5.36)	0.08040(3.87)

TABLE III: L2 error of NOs in predicting numerically simulated results of ARZ equation with different initial conditions

simulating systems with clear, distinct states or zones and tests each NO's effectiveness in handling discontinuities.

Mixed This set combines various types of functions mentioned above to create a complex and diverse testing environment. It aims to challenge the learning capabilities of each NO in simultaneously handling a wide range of initial conditions.

We found that the Lax-Friedrichs scheme will be unstable when solving PDEs with piece-wise constant initial condition when the grid is space. Therefor, we excluded the piece-wise constant dataset for grid size of 2^7.

C. Training

The primary goal of this paper is to assess the performance of various neural operators (NOs) across a diverse range of data, from smooth to sharp, discontinuous or high frequency functions and from simulated to real datasets. To ensure a fair comparison, we meticulously adhere to the training protocols outlined in the original papers. However, given the multi-channel inputs/outputs and complex boundary conditions in our dataset, we made essential modifications to accommodate these factors.

For DeepONet and POD-DeepONet, modifying the network to handle different types of inputs is straightforward. The Branch Net, which processes initial and boundary conditions, essentially comprises a fully connected network that accepts flattened vectors. To incorporate various inputs, simply flatten each input and concatenate them. For the outputs, both DeepONet and POD-DeepONet inherently support 2D or even 3D outputs. Users need only specify the points of interest (e.g., time t and position x in a 2D space) for evaluation, which are then fed to the Trunk Net or the POD Basis. To enable multi-channel 2D output, an additional input c, with binary values 0 or 1, indicates the channel being evaluated as shown in Fig 1.

The FNO (Fourier Neural Operator) family naturally supports multi-channel input and output. In this architecture, an uplifting neural network P projects input channels to a higher-dimensional latent representation, while another network Q reduces these dimensions back to the desired output channels. However, the FNO family requires that all channels maintain the same dimensions, necessitating careful input design. In our approach, to integrate initial conditions with the FNO family, we replicate the initial condition across the time axis to form a matrix matching the output's shape. Similarly, for boundary conditions, we replicate these along the spatial axis to achieve the same effect. These methods are demonstrated in Fig2.

We did not train MWT with NGSIM data due to the specific design constraints of MWT, which requires input dimensions to be powers of two. Although it is feasible to interpolate real-world data to meet this requirement, determining the appropriate interpolation method poses a challenge, especially since our objective is to compare different neural operators (NOs). Interpolation outcomes could potentially benefit one NO's performance while impairing another's. Consequently, we chose to exclude MWT from our comparison in NGSIM dataset and focused solely on evaluating FNO, TFNO, DeepONet, and POD-DeepONet.

V. Results

A. Results on NGSIM Dataset

The results for training NOs with the NGSIM dataset are detailed in Table I. Overall, TFNO outperformed all other models, while FNO exhibited the weakest performance. POD-DeepONet slightly surpassed DeepONet, which generated overly smooth results that poorly aligned with the actual data, as shown in Fig 3. The data in Table I further reveals that training with upstream and downstream traffic flow information (used as boundary conditions) significantly

enhanced the prediction accuracy of all neural operators (NOs), particularly with downstream traffic flow. However, FNO benefited the least from the additional downstream flow data, finding also Reflected in Fig 3. To further explore the importance of both upstream and downstream traffic flow information in traffic flow estimation, we included a visualization of TFNO's predictions under various initial and boundary condition combinations in Fig 4. This visualization corroborates the main findings from Table I, indicating minimal difference with upstream traffic flow information alone, whereas predictions with downstream information align more closely with the ground truth.

B. Results on Simulated Traffic Flow Datasets

The results from the simulated traffic flow datasets are presented in Tables II and III. Overall, MWT performed best in relatively smooth datasets generated using a sparse grid in the Lax-Friedrichs method, while TFNO excelled in other scenarios. For the single-channel datasets based on the LWR equation, FNO and DeepONet were the least effective, particularly in datasets with mixed initial conditions. This highlights the improvements in subsequent versions of these networks, enhancing their capacity to handle diverse initial conditions.

In multi-channel datasets, FNO's performance was comparable to TFNO and MWT, surpassing both DeepONet and POD-DeepONet. The general architecture of the FNO family, shown in Fig 2, appears to manage well-structured multi-channel data more effectively than the DeepONet family. Interestingly, despite the common belief that FNO struggles with piece-wise constant data due to high frequency errors around the discontinuities when using Fourier Series, our results show that the FNO family still outperforms the DeepONet family in these scenarios. Moreover, all members of the FNO family had similar performances on piece-wise constant data, suggesting that the basic design of FNO handles these types of data most effectively.

Finally, POD-DeepONet consistently outperformed DeepONet, a finding also observed in the NGSIM data. This improvement for DeepONet is likely due to the use of a proper orthogonal decomposition basis derived from data through PCA prior to training, enhancing its overall performance.

VI. Conclusion

In this paper, we conducted a comprehensive evaluation of various Neural Operators (NOs) for Traffic State Estimation (TSE). Our findings reveal that while some NOs like MWT and TFNO excel in some use case, they do require more complex input preparations, particularly MWT. In contrast, the DeepONet family allows for simpler input integration, making it more user-friendly and less demanding of machine learning expertise. We also addressed a prevalent misconception in the field: that the FNO family struggles with piece-wise constant data compare to other NOs due to high-frequency errors from Fourier Series approximations. Our results prove otherwise. Additionally, we underscored the importance of including both upstream and downstream traffic information for effective TSE, a common sense well-established in the field of transportation. Consequently, the choice of the best NO for TSE is not straightforward; it should be base on the specific requirements and capabilities of the user. This principle likely extends to other applications of NOs. We hope this work not only aids users in selecting the ideal NO for their needs but also inspire further research in addressing the disadvantage of each NOs and the develope of new NOs.

References

[1] T. Seo, A. M. Bayen, T. Kusakabe, and Y. Asakura, "Traffic state estimation on highway: A comprehensive survey," *Annual reviews in control*, vol. 43, pp. 128–151, 2017.

[2] X. Di, R. Shi, Z. Mo, and Y. Fu, "Physics-informed deep learning for traffic state estimation: A survey and the outlook," *Algorithms*, vol. 16, no. 6, p. 305, 2023.

[3] R. Shi, Z. Mo, K. Huang, X. Di, and Q. Du, "A physics-informed deep learning paradigm for traffic state and fundamental diagram estimation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 11 688–11 698, 2021.

[4] L. Lu, P. Jin, and G. E. Karniadakis, "Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators," *arXiv preprint arXiv:1910.03193*, 2019.

[5] L. Lu, X. Meng, S. Cai, Z. Mao, S. Goswami, Z. Zhang, and G. E. Karniadakis, "A comprehensive and fair comparison of two neural operators (with practical extensions) based on fair data," *Computer Methods in Applied Mechanics and Engineering*, vol. 393, p. 114778, 2022.

[6] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Fourier neural operator for parametric partial differential equations," *arXiv preprint arXiv:2010.08895*, 2020.

[7] J. Kossaifi, N. Kovachki, K. Azizzadenesheli, and A. Anandkumar, "Multi-grid tensorized fourier neural operator for high-resolution pdes," *arXiv preprint arXiv:2310.00120*, 2023.

[8] G. Gupta, X. Xiao, and P. Bogdan, "Multiwavelet-based operator learning for differential equations," *Advances in neural information processing systems*, vol. 34, pp. 24 048–24 062, 2021.

[9] X. Chen, F. Yongjie, S. Liu, and X. Di, "Physics-informed neural operator for coupled forward-backward partial differential equations," in *1st Workshop on the Synergy of Scientific and Machine Learning Modeling@ ICML2023*, 2023.

[10] B. T. Thodi, S. V. R. Ambadipudi, and S. E. Jabari, "Fourier neural operator for learning solutions to macroscopic traffic flow models: Application to the forward and inverse problems," *Transportation Research Part C: Emerging Technologies*, vol. 160, p. 104500, 2024.

[11] P. I. Richards, "Shock waves on the highway," *Operations research*, vol. 4, no. 1, pp. 42–51, 1956.

[12] M. J. Lighthill and G. B. Whitham, "On kinematic waves ii. a theory of traffic flow on long crowded roads," *Proceedings of the royal society of london. series a. mathematical and physical sciences*, vol. 229, no. 1178, pp. 317–345, 1955.

[13] C. F. Daganzo, "The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory," *Transportation research part B: methodological*, vol. 28, no. 4, pp. 269–287, 1994.

[14] A. Aw and M. Rascle, "Resurrection of" second order" models of traffic flow," *SIAM journal on applied mathematics*, vol. 60, no. 3, pp. 916–938, 2000.

[15] H. M. Zhang, "A non-equilibrium traffic model devoid of gas-like behavior," *Transportation Research Part B: Methodological*, vol. 36, no. 3, pp. 275–290, 2002.

[16] T. Chen and H. Chen, "Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems," *IEEE transactions on neural networks*, vol. 6, no. 4, pp. 911–917, 1995.

[17] U.S. Department of Transportation, "Next generation simulation (ngsim) dataset," U.S. Department of Transportation Intelligent Transportation Systems Joint Program Office, 2007. [Online]. Available: https://ops.fhwa.dot.gov/trafficanalysistools/ngsim.htm

[18] R. Shi, Z. Mo, and X. Di, "Physics-informed deep learning for traffic state estimation: A hybrid paradigm informed by second-order traffic models," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 1, 2021, pp. 540–547.

[19] Z. Mo, Y. Fu, D. Xu, and X. Di, "Trafficflowgan: Physics-informed flow based generative adversarial network for uncertainty quantification," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2022, pp. 323–339.