Neural Network Enhanced Shock-Capturing Schemes for Compressible Flows

Yung-Tien Lin*
Department of Mechanical and Aerospace Engineering, UCLA
420 Westwood Plaza, Los Angeles, CA
(Dated: June 6, 2022)

In this research, machine learning techniques will be applied to shock-capturing schemes to further enhance the numerical method accuracy on unsteady turbulent compressible flow problems. The main purpose of this research is to find a more computationally efficient method for evaluating fluxes through artificial neural networks. To reduce the cost of the data-driven method, the neural network has to be compact, physics-informed, and intrinsically numerical accurate. In this research, the target shock-capturing method is the fifth-order weighted essentially non-oscillatory (WENO) method. The flux evaluation coefficient will be replaced by a neural network and trained by several analytical functions. The testing problems are the linear advection equation and the Shu-Osher problem. These problems consist of discontinuous shock interfaces and some of them are turbulent where traditional WENO methods would introduce too much numerical diffusion.

I. INTRODUCTION

High-order CFD methods are one of the most popular research topics in the CFD research community. The methods are potentially more computationally efficient than the low-order methods by using fewer computational grid points to achieve the same degree of accuracy [1]. However, due to the Gibbs phenomenon, conventional high-order approximations will suffer numerical instability in the vicinity of the discontinuous interfaces and eventually lead to failed simulations. This causes high-order methods very hard to apply to complex application problems, especially for flow fields that are highly nonlinear, turbulent, or contain shock waves. In the field of supersonic and hypersonic simulation, shock-fitting and shock-capturing methods are the two common approaches to address problems involved with strong shock waves.

The shock-fitting method is introduced by Moretti and Abbett [2] for simulating the supersonic flow around blunt objects. The method treats the shock front as one of the boundary conditions and the inflow states are computed through the Rankine-Hugoniot jump condition [3]. To track the location of the shock front, the shock movement can be computed through the time derivative of the Rankine–Hugoniot conditions with a characteristic relation [4], the time derivative of the Rankine–Hugoniot conditions with a momentum equation [5], or the shock velocity evaluated through Riemann invariants [2, 6]. Since the fluid states behind the shock front usually are relatively smooth compared with the solution around the shock, this allows conventional high-order approaches can be applied to the problem with strong shock when combined with shock-fitting techniques [7]. However, shock-fitting requires posteriori knowledge of the flow field to specify the shock front location in the computational domain. This makes shock-fitting methods are hard to apply on problems with multiple shock interactions or flow field with shock bubbles.

On the other hand, shock capturing methods use more robust flux evaluation schemes and thus require no empirical understanding of the flow field. Hence shock-capturing methods are more popular in both academic research and industrial applications for the past few decades [7]. To address the numerical instability due to the strong shock, several shock-capturing frameworks have been proposed, such as the total variation diminishing (TVD) approach from Harten [8], Monotonic Upstream-centered Schemes for Conservation Laws (MUSCL) from Van Leer [9], and the essentially non-oscillatory (ENO) family from Harten [10]. In the ENO family, the weighted essentially non-oscillatory (WENO) [11–13] methods gain more popularity among other shock-capturing methods due to the adaptive stencil selection process with nonlinear weights. Even though WENO methods have shown their capability in hypersonic flow simulations [14–16], the methods introduce excess amounts of numerical diffusion around discontinuous interfaces [17, 18]. To further improve the accuracy and efficiency of the WENO methods, several WENO variant schemes are proposed and tested on different PDE problems [19].

In recent years, data-driven modeling and analysis have become popular research topics in the field of fluid mechanics [20, 21]. The underlying physics can be extracted from analyzing the data of large-scale simulation or experimental measurement and helps researchers to understand fluid behavior in different ways. Data-driven techniques have been applied to PDE solving for accelerating simulations or improving solution accuracy. One of the common approaches is implementing an artificial neural network into the simulation routing and using the output from the network to estimate numerical coefficients or variables. To ensure the conservation laws of fluid, most of the data-driven models are physics-informed. One of the application is applying neural network on turbulence modeling for evaluating the anisotropic Reynolds stress [22], LES wall shear stress

* yungtlin@ucla.edu

[23], or subgrid-scale eddy viscosity [24].

This research focuses on applying neural network on the shock-capturing scheme, or more specific, the WENO methods from Jiang and Shu [12]. Similar works have been done by literatures [25–27].

II. OBJECTIVES

The goal of this research is to improve the 5th-order WENO method with neural networks in terms of computational efficiency. The idea is to construct a physic-informed model for the compressible flow problems. The ultimate goal is to establish a framework for effectively designing neural networks which can be applied to various nonlinear PDE simulations, to accelerate the simulation of compressible flow, and to discover undiscovered numerical properties for CFD.

III. APPROACH

A. Weighted Essentially Non-Oscillatory Scheme

The one-dimensional scalar conservation law can be expressed as follow:

$$\frac{\partial u}{\partial t} + \frac{\partial f}{\partial x} = 0 \tag{1}$$

u is the state variable and f is the flux for the corresponding state. For the essentially non-oscillatory (ENO) family, the methods consider the solution on the grids are cell average of the state $\bar{u}$. The conservation law of cell average can be obtained through volume integrating equation 1 with a given domain $\Omega = \{(x,t) : x_{j-1/2} \leq x \leq x_{j+1/2}, t_i \leq t \leq t_{i+1}\}$ and apply Green's theorem:

$$\int_{x_{j-1/2}}^{x_{j+1/2}} \left(u^{i+1} - u^i\right) dx + \int_{t_i}^{t_{i+1}} \left(f_{j+1/2} - f_{j-1/2}\right) dt = 0 \tag{2}$$

note the cell average:

$$\bar{u}_j = \int_{x_{j-1/2}}^{x_{j+1/2}} u \, dx \tag{3}$$

with an infinitismally small time step and the grid size h, the time derivative of the cell average:

$$\frac{\partial \bar{u}}{\partial t} + \frac{1}{h}\left(f_{j+1/2} - f_{j-1/2}\right) = 0 \tag{4}$$

The above equation provides the framework of the ENO family approach. The equation is similar to the finite volume method but also can be used in finite difference method.

The challenge of cell average representation is to estimate the flux of the actual state on the cell edge from cell averages. The flux can utilize the neighboring cell averages and be estimated by the reconstruction via deconvolution [10] or the reconstruction via primitive function [11] process. For instance, the possible formulations of third-order ($r = 3$) flux reconstruction on a cell edge can be:

$$\begin{aligned}
\hat{f}^{(1)}_{j+1/2} &= \frac{1}{3}\bar{f}_{j-2} - \frac{7}{6}\bar{f}_{j-1} + \frac{11}{6}\bar{f}_j \\
\hat{f}^{(2)}_{j+1/2} &= -\frac{1}{6}\bar{f}_{j-1} + \frac{5}{6}\bar{f}_j + \frac{1}{3}\bar{f}_{j+1} \\
\hat{f}^{(3)}_{j+1/2} &= \frac{1}{3}\bar{f}_j + \frac{5}{6}\bar{f}_{j+1} - \frac{1}{6}\bar{f}_{j+2}
\end{aligned} \tag{5}$$

where $\bar{f}$ is the flux based on cell average on the grid.

One of the novelties in the ENO family schemes is the algorithm adaptively selecting stencils depending on the total variation. In Harten approach [8], they measure the total variation of candidate stencils recursively in L^1-norm and select the smoothest stencil. Shu and Osher [11] measure the total variation in L^2-norm which the measurement coefficients can be precomputed and the recursive stencil searching process is not required. Liu et al. [28] introduced the first weighted essentially non-oscillatory (WENO) scheme in the literature which uses nonlinear weights ω to linearly combine the candidate stencils together. In other words, the WENO method approximates the reconstructed flux on cell edges by a summation of the candidate stencils ($r = 3$):

$$f_{j+1/2} \approx \hat{f}_{j+1/2} = \omega_1 \hat{f}^{(1)}_{j+1/2} + \omega_2 \hat{f}^{(2)}_{j+1/2} + \omega_3 \hat{f}^{(3)}_{j+1/2} \tag{6}$$

where

$$\omega_i = \frac{\sigma_i}{\sum_{k=1}^{r} \sigma_k} \tag{7}$$

and

$$\sigma_i = \frac{\gamma_i}{(\epsilon^* + \beta_i)^2} \tag{8}$$

$\epsilon^* = 10^{-6}$ and γ is the optimal coefficients of the scheme. For $r = 3$ case, the optimal coefficients are:

$$\gamma_1 = \frac{1}{10}, \qquad \gamma_2 = \frac{3}{5}, \qquad \gamma_3 = \frac{3}{10} \tag{9}$$

When the smooth indicators β are all identical or small enough, the nonlinear weights ω will reduce to the optimal coefficients γ which make the flux reconstruction process $2r-1$th order accurate. Hence, the WENO method with $r = 3$ is also known as WENO5. Jiang and Shu [12] suggested the smooth indicator β should be computed as follow to guarantee the $2r-1$th order accuracy in smooth region:

$$\beta_i = \sum_{l=1}^{r-1} h^{2l-1} \int_{x_{j-1/2}}^{x_{j+1/2}} \left(\frac{\partial^l \hat{f}^{(i)}}{\partial \xi^l}\right)^2 d\xi \tag{10}$$

For $r = 3$ case, the above integration yields:

$$\begin{aligned}
\beta_1 &= \frac{13}{12}(\bar{f}_{j-2} - 2\bar{f}_{j-1} + \bar{f}_j)^2 + \frac{1}{4}(\bar{f}_{j-2} - 4\bar{f}_{j-1} + 3\bar{f}_j)^2 \\
\beta_2 &= \frac{13}{12}(\bar{f}_{j-1} - 2\bar{f}_j + \bar{f}_{j+1})^2 + \frac{1}{4}(\bar{f}_{j-1} - \bar{f}_{j+1})^2 \\
\beta_3 &= \frac{13}{12}(\bar{f}_j - 2\bar{f}_{j+1} + \bar{f}_{j+2})^2 + \frac{1}{4}(3\bar{f}_j - 4\bar{f}_{j+1} + \bar{f}_{j+2})^2
\end{aligned} \tag{11}$$

This study will focus on the WENO method ($r = 3$) implemented by Jiang and Shu [12] and the scheme will be referred to WENO5-JS. The following neural network enhanced shock-capturing schemes will base on the WENO5-JS.

B. Stevens and Colonius Approach

In Stevens and Colonius research [25], they proposed a methodology to enhance WENO5-JS and potentially other shock-capturing schemes. The idea of this approach is to utilize the coefficients from a known shock-capturing method, in this case, WENO5-JS, and improve the coefficients for evaluating the flux on cell edge through a neural network. They expressed equation 6 in terms of coefficients of cell average fluxes:

$$\hat{f}_{j+1/2} = c_{-2}\bar{f}_{j-2} + c_{-1}\bar{f}_{j-1} + c_0\bar{f}_j + c_1\bar{f}_{j+1} + c_2\bar{f}_{j+2} \tag{12}$$

The first step of obtaining the neural network enhanced coefficients, c_i, is computing the corresponding coefficients from WENO5-JS, $\tilde{c}_i$:

$$\begin{aligned}
\tilde{c}_{-2} &= \frac{1}{3}\omega_1 \\
\tilde{c}_{-1} &= -\frac{7}{6}\omega_1 - \frac{1}{6}\omega_2 \\
\tilde{c}_0 &= \frac{11}{6}\omega_1 + \frac{5}{6}\omega_2 + \frac{1}{3}\omega_3 \\
\tilde{c}_1 &= \frac{1}{3}\omega_2 + \frac{5}{6}\omega_3 \\
\tilde{c}_2 &= -\frac{1}{6}\omega_3
\end{aligned} \tag{13}$$

Then these five WENO5-JS coefficients are the input to the neural network which returns the change to each coefficient, $\Delta\tilde{c}_i$. The network has three hidden layers with three perceptrons in each layer. The study of neural network structure will be discussed in the later section. After the change of the coefficients, $\Delta\tilde{c}_i$, have been decided, the value will be added to the WENO5-JS coefficients, $\tilde{c}_i$, to get the preliminary coefficients, $\hat{c}_i$. To ensure the 1st-order interpolation accuracy, the sum of the resulting coefficients, c, should be unity. This can be achieved by converting the constraint to a problem that minimizes the change of coefficient in terms of L^2-norm:

$$\begin{aligned}
&\min_{c\in\mathbb{R}^5} \sum_{i=-2}^{2} (c_i - \hat{c}_i) \\
&\text{s.t.} \sum c_i = 1
\end{aligned} \tag{14}$$

where c is the finalized machine learning coefficients. The above affine transformation can be analytically computed by:

$$\Delta c_i = \frac{1 - \sum \hat{c}_i}{5} \tag{15}$$

and finally:

$$c_i = \hat{c}_i + \Delta c_i \tag{16}$$

The neural network predicted flux, $\hat{f}$, is computed by an inner product of the finalized coefficients, c_i, with the given cell average flux, $\bar{f}$. FIG. 1 is the resulting schematic of the entire prediction procedure. In the literature, Stevens and Colonius referred to this network as WENO-NN. The details of the training data generation for the networks will be discussed in the next section.

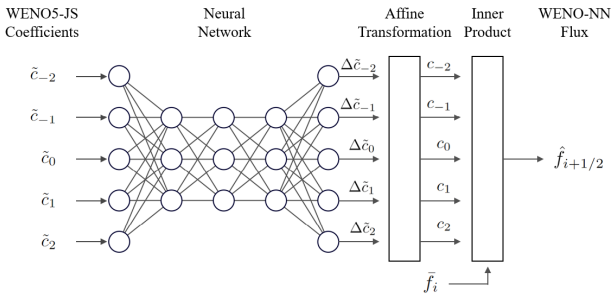

FIG. 1. Schematic of the neural network enhanced shock-capturing procedure by Stevens and Colonius [25]

IV. DATA SET

A. Training Data

The training data for WENO-NN is from a set of analytic functions. The input cell average data is obtained through the spatial integration of given functions using equation 3 and the output of the flux on the cell edge will also be stored. The candidate functions are step functions, sawtooth waves, hyperbolic tangent functions, sinusoids, polynomials, and sums of the above. FIG. 2 shows a few couple data points from candidate functions that are used in the training process. The sampling rate of the data set is at least two times the Nyquist sampling rate of periodic waves to avoid the data set being contaminated by signal aliasing. The overall candidate

function distribution in the training data set is shown in FIG. 3. To prevent redundant information in the data set, every new entry will be checked if any similar data already existed. The data checking algorithm is based on finding the nearest neighbor using k-d tree [29] and computes the data point distance in L_2 norm sense. The amount of data entries is around 80,000.

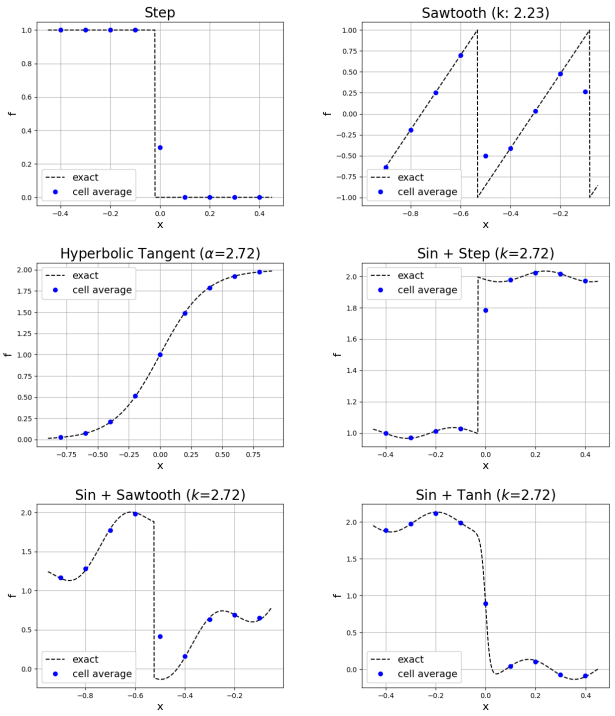

FIG. 2. Candidate functions for the neural network training process

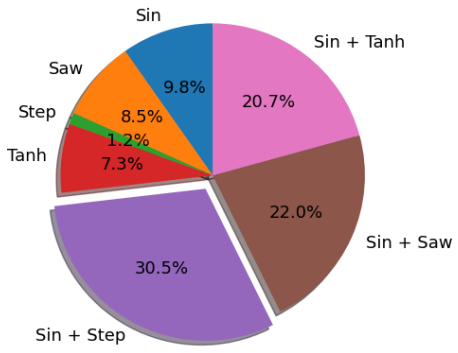

FIG. 3. The candidate function distribution in the training data set

V. NUMERICAL IMPLEMENTATIONS

A. WENO5-JS Solver

The WENO5-JS solver is the base solver for all of the data-driven approaches and coded in C++. The solver establishes the fundamental framework for implementing machine learning shock-capturing schemes and provides baseline solutions for evaluating computational efficiency between different methods. For the flux splitting, the solver uses the local Lax-Friedrichs method [4] for low computation time and an acceptable amount of numerical diffusion:

$$\begin{aligned} f^{\pm}_{LF}(u) &= \frac{1}{2}\left[f(u) + \pm\alpha u\right] \\ \alpha &= \max\left|\frac{\partial f}{\partial u}\right| \end{aligned} \tag{17}$$

where α is the maximum eigenvalue from the system Jacobian matrix. The third-order SSP Runge-Kutta scheme [30] is used for time integration for the temporal total variation diminishing property and the low storage character. The time-stepping procedure can be expressed as:

$$\begin{aligned} u^{(1)} &= u^{(n)} + \Delta t L(u^{(n)}) \\ u^{(2)} &= \frac{3}{4}u^{(n)} + \frac{1}{4}u^{(1)} + \frac{1}{4}\Delta t L(u^{(1)}) \\ u^{(n+1)} &= \frac{1}{3}u^{(n)} + \frac{2}{3}u^{(2)} + \frac{2}{3}\Delta t L(u^{(2)}) \end{aligned} \tag{18}$$

The validity of the WENO5-JS solver has been verified by the Shu-Osher problem. Shu-Osher problem is a classic test case with Euler equations for shock-capturing due to the problem contains both shock wave and turbulent behavior. FIG. 4 shows the density distribution from both Shu and Osher [11] and the implemented WENO5-JS solver. The exact solution is computed from 10,000 uniform grid points in x-direction with CFL number $\nu = 0.8$. The numerical results are based on 200 gird points. The literature and the WENO5-JS are consistent with each other. Both low-resolution results are smeared right behind the shock due to the excess amounts of numerical error in the ENO family. The consistency of the two figures provides reliability to the WENO5-JS solver.

B. WENO-NN

WENO-NN uses both Keras [31] and TensorFlow2.0 [32] package in Python for the network training.The network weights are searched through the Adam optimizer [33]. Adam optimizer uses stochastic gradient descent to prevent neural network traps into local minimum and records the descending direction to simulate the momentum of gradient drop. To make the training process parallelizable, batch training is used for the training process. The data is split into batches with around 80 data points

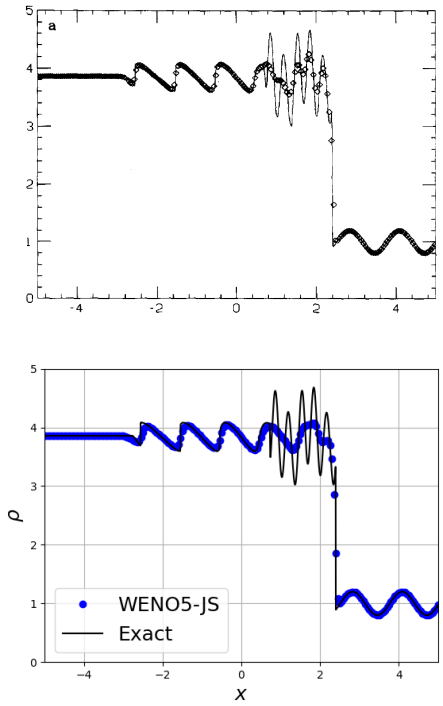

FIG. 4. Density distribution of Shu-Osher problem from Shu and Osher [11] (top) and the implemented WENO5-JS solver (bottom)

in each batch. The training process requires around 10 epochs to complete. Further training on the model has no significant change on both training and validation losses. The weights and bias of the network are randomly initialized. Rectified linear unit (ReLU) function is selected for the hidden layer activation function and the output activation function is a linear function. To prevent network over-fitting the data set and reducing total variation in the simulation, L_2 regularization is used and only applied to the output of the neural network. The constraint on the output layer makes the model return non-zero values only if the coefficient change is necessary. Hence, the regularisation factor is significant in the resulting model and the detailed study of the factor will be discussed in the followed chapter. The input data set is split into a training set of 80% of the data and a validation set of the other 20%. Mean squared error (MSE) is used for the training loss:

$$MSE = \frac{\sum_i^N (y_i - y_i^*)^2}{N} \tag{19}$$

FIG. 5 is the mean squared error history when the training process epoch is set to 50. Since the model is small and uses a high regularization factor, the model will converge fast and the output will be less likely over-fitting the data set. However, this will also restrict the flexibility of network output which makes it harder to be trained. To prevent the network is over-train, the training process is forced stopped at epochs 10.

FIG. 5. RMSE history of the training process for 50 epochs

VI. RESULTS

A. Training Set

FIG. 6. shows the prediction results of WENO5-JS and the WENO-NN. Also, the figure displays the influence of output layer regularization on the neural network results. The top figure is trained without regularization and the below figure with regularization factor $\lambda = 3$. In both of the figures, WENO5-JS fails to predict the flux in the vicinity of the discontinuous interface. This makes the blue dots distributed in a mirrored "N" shape. To improve the WENO scheme, the neural network should help the shock-capturing estimate the flux near the solution jump more accurately. The ideal scatter plot will be both vertical lines shorten. In the non-regularization case, the model's overall performance is better than the regularized model in terms of training loss. The data points of the non-regularized model are closer to the exact solutions. However, the non-regularized model tends to over-extrapolating the solution by predicting the flux over 1 or below 0. This property will make the resulting scheme generate more total variation (TV) in the solution and more easily lead to a diverged simulation. On the other hand, the regularization technique reduces the amount of over-extrapolated data in exchange for the overall accuracy. Hence, the regularization factor should be large enough to have a stable result but also small as possible to increase the prediction accuracy.

B. Linear Advection

The first simulation test is the linear advection problem with a square wave. This problem is simple but also challenging to the high-order shock-capturing schemes. The discontinuous interfaces in the square wave will cause the high-order scheme to oscillate and potentially lead to a diverged solution. Even for WENO5-JS, it still generates a small amount of total variation in the square wave. Further, the scheme also dissipates the solution near the

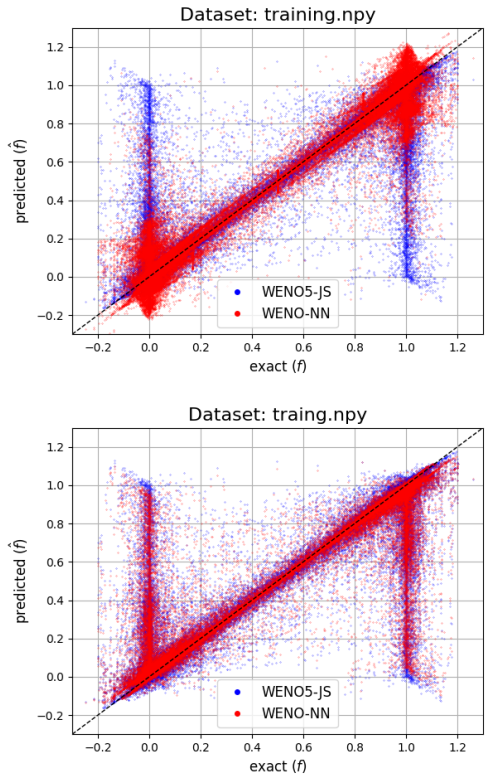

FIG. 6. The scatter plots of predicted versus exact solution with different regularization factor $\lambda = 0$ (top) and $\lambda = 3$ (bottom)

solution jump. Hence, this study looks for the neural network to improve the simulation in terms of accuracy and total variation. To better compare the simulation results, the total variation (TV) is measured as below:

$$TV = \sum_i |u_i - u_{i-1}| \tag{20}$$

FIG. 7 and FIG. 8 show the training results with different parameters used. For the regularization factor, models with high constraint values behave similarly to WENO5-JS and the change in outputs is limited. For low regularization training, results have more flexibility but most of the models will lead to diverged simulation. Hence, an ideal selection of regularization factors is high enough to obtain a TV conserved model but low as possible to improve the accuracy. On the hand, neurons per layer have a small impact on the simulation when the neuron number is larger than 3. This is probably due to the input dimension being small and limited information can be used. In general, the scatter plots show that the model performance usually is a trade-off between simulation accuracy and total variation generation.

FIG. 9 shows the square wave advection results with different degree of TV in WENO-NN models. The wave speed, c, of the problem is 1 and simulated with CFL number $\nu = 0.5$. Three different levels of TV are included

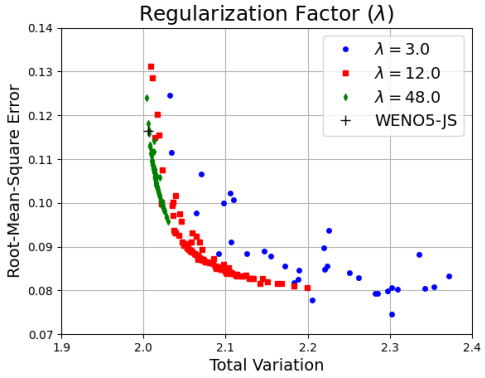

FIG. 7. Model performance scatter plot with different regularization factors used

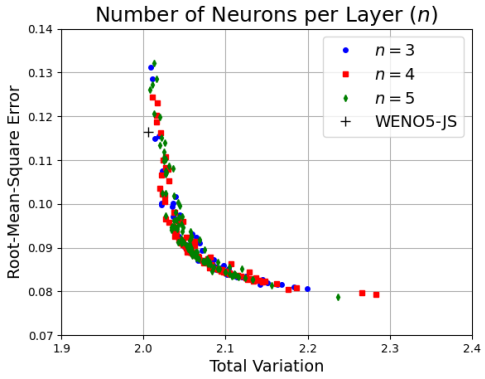

FIG. 8. Model performance scatter plot with different neurons used in the network

in the figure from low (2.0017) to medium (2.0095) to high (2.2056). Like most low-order numerical methods, low TV approaches conserve the fluctuation inside the simulation but smear out the solution sharpness. This character guarantees a successful run of the simulation with the exchange of simulation accuracy. On the other hand, high TV approaches have better accuracy but introduce oscillations to the problem. These over-shoot results will continue growing in the simulation and eventually leads to a diverged solution. Hence an ideal WENO-NN model should improves the simulation but generate a limited amounts of TV.

C. Shu-Osher Problem

The initial and boundary conditions of the problem are given in the study from Shu and Osher [11]. In this simulation, if a WENO-NN is trained without regularization on the output layer, the simulation will diverge immediately after a few time steps. With a moderate amount of regularization factor used, the simulation can be used in the simulation and an improved simulation results can be obtained. FIG. 10. shows re-

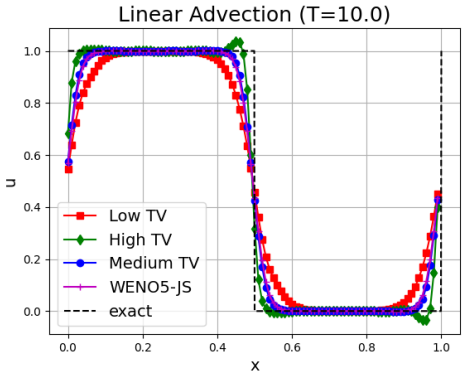

FIG. 9. WENO-NN on the linear advection problem with different degrees of total variation (TV) in the model

sults of WENO5-JS and WENO-NN when apply to the Shu-Osher problem. In WENO5-JS, the shock-capturing scheme fails to accurately predict the nonlinear advection wave propagation when the high-frequency oscillation collapsed. In WENO-NN result, the lower frequency waves are captured more accurately and waves are sharper. Even though WENO-NN is not performed extraordinarily good in the linear advection problem, the model is still able to improve the simulation in Shu-Osher problem. This might due to training process assists the model to understand the flow transition better. In FIG. 11, WENO-NN is slightly better than WENO5-JS in terms of L_1 error.

VII. DISCUSSION AND CONCLUSION

This study shows that the machine learning technique can be used for improving compressible flow simulation with an appropriate design of the model. To apply a neural network to a simulation framework, the model has to consider various numerical properties in the model design, such as numerical accuracy and total variation diminishing. One of the approaches is utilizing a known scheme and building a network training process based on the method. Even though the resulting neural network model may have limited change in the output, those small changes will accumulate and eventually lead to an observable improvement over time. Also, this framework provides a general model which also can be used in flow fields other than the flow fields mentioned in this study. However, this study does not perfectly reproduce the results from Stevens and Colonius [25] research. Their model has better accuracy even in the high-frequency region. Possible ways to achieve the result can be having a broader model studying or tuning the training process. Overall, machine learning is a robust tool to construct a data-driven scheme when a right data set and model design are given.

All of the code implementations for this project are uploaded to the GitHub repository: https://github.com/yungtlin/WENO-NN.

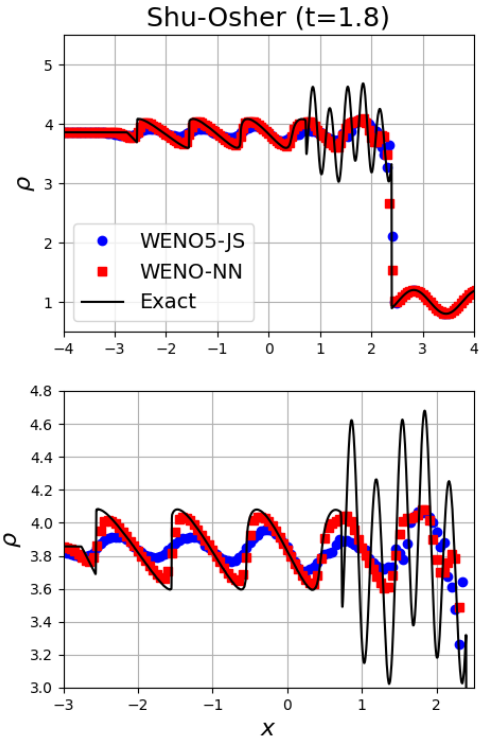

FIG. 10. Shu-Osher problem with shock-fitting scheme WENO5-JS and WENO-NN ($\lambda = 12$)

[1] Z. J. Wang, K. Fidkowski, R. Abgrall, F. Bassi, D. Caraeni, A. Cary, H. Deconinck, R. Hartmann, K. Hillewaert, H. T. Huynh, *et al.*, High-order cfd methods: current status and perspective, International Journal for Numerical Methods in Fluids **72**, 811 (2013).
[2] G. Moretti and M. Abbett, A time-dependent computational method for blunt body flows., AIAA Journal **4**, 2136 (1966).
[3] J. D. Anderson, *Modern compressible flow: with historical perspective*, Vol. 12 (McGraw-Hill New York, 1990).
[4] X. Zhong, High-order finite-difference schemes for numerical simulation of hypersonic boundary-layer transition, Journal of Computational Physics **144**, 662 (1998).
[5] A. K. Henrick, T. D. Aslam, and J. M. Powers, Simulations of pulsating one-dimensional detonations with true fifth order accuracy, Journal of Computational Physics **213**, 311 (2006).
[6] G. Moretti, The λ-scheme, Computers & Fluids **7**, 191

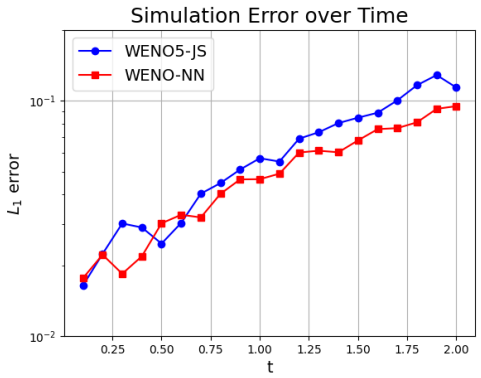

FIG. 11. Simulation L_1 error over time in Shu-Osher problem

(1979).
[7] P. S. Rawat and X. Zhong, On high-order shock-fitting and front-tracking schemes for numerical simulation of shock–disturbance interactions, Journal of Computational Physics **229**, 6744 (2010).
[8] A. Harten, High resolution schemes for hyperbolic conservation laws, Journal of computational physics **135**, 260 (1997).
[9] B. Van Leer, Towards the ultimate conservative difference scheme. v. a second-order sequel to godunov's method, Journal of computational Physics **32**, 101 (1979).
[10] A. Harten, B. Engquist, S. Osher, and S. R. Chakravarthy, Uniformly high order accurate essentially non-oscillatory schemes, iii, in *Upwind and high-resolution schemes* (Springer, 1987) pp. 218–290.
[11] C.-W. Shu and S. Osher, Efficient implementation of essentially non-oscillatory shock-capturing schemes, Journal of computational physics **77**, 439 (1988).
[12] G.-S. Jiang and C.-W. Shu, Efficient implementation of weighted eno schemes, Journal of computational physics **126**, 202 (1996).
[13] C.-W. Shu, High order weighted essentially nonoscillatory schemes for convection dominated problems, SIAM review **51**, 82 (2009).
[14] L. Duan, M. M. Choudhari, and M. Wu, Numerical study of acoustic radiation due to a supersonic turbulent boundary layer, Journal of Fluid Mechanics **746**, 165 (2014).
[15] C. Zhang, L. Duan, and M. M. Choudhari, Direct numerical simulation database for supersonic and hypersonic turbulent boundary layers, AIAA journal **56**, 4297 (2018).
[16] X. Li, D. Fu, and Y. Ma, Direct numerical simulation of hypersonic boundary layer transition over a blunt cone, AIAA journal **46**, 2899 (2008).
[17] E. Johnsen, J. Larsson, A. V. Bhagatwala, W. H. Cabot, P. Moin, B. J. Olson, P. S. Rawat, S. K. Shankar, B. Sjögreen, H. C. Yee, *et al.*, Assessment of high-resolution methods for numerical simulations of compressible turbulence with shock waves, Journal of Computational Physics **229**, 1213 (2010).
[18] C. Brehm, M. F. Barad, J. A. Housman, and C. C. Kiris, A comparison of higher-order finite-difference shock capturing schemes, Computers & Fluids **122**, 184 (2015).
[19] C.-W. Shu, High order weno and dg methods for time-dependent convection-dominated pdes: A brief survey of several recent developments, Journal of Computational Physics **316**, 598 (2016).
[20] S. L. Brunton, B. R. Noack, and P. Koumoutsakos, Machine learning for fluid mechanics, Annual Review of Fluid Mechanics **52**, 477 (2020).
[21] K. Duraisamy, G. Iaccarino, and H. Xiao, Turbulence modeling in the age of data, Annual Review of Fluid Mechanics **51**, 357 (2019).
[22] J. Ling, A. Kurzawski, and J. Templeton, Reynolds averaged turbulence modelling using deep neural networks with embedded invariance, Journal of Fluid Mechanics **807**, 155 (2016).
[23] X. Yang, S. Zafar, J.-X. Wang, and H. Xiao, Predictive large-eddy-simulation wall modeling via physics-informed neural networks, Physical Review Fluids **4**, 034602 (2019).
[24] Z. Wang, K. Luo, D. Li, J. Tan, and J. Fan, Investigations of data-driven closure for subgrid-scale stress in large-eddy simulation, Physics of Fluids **30**, 125101 (2018).
[25] B. Stevens and T. Colonius, Enhancement of shock-capturing methods via machine learning, Theoretical and Computational Fluid Dynamics **34**, 483 (2020).
[26] T. Kossaczká, M. Ehrhardt, and M. Günther, Enhanced fifth order weno shock-capturing schemes with deep learning, Results in Applied Mathematics **12**, 100201 (2021).
[27] D. A. Bezgin, S. J. Schmidt, and N. A. Adams, Weno3-nn: A maximum-order three-point data-driven weighted essentially non-oscillatory scheme, Journal of Computational Physics **452**, 110920 (2022).
[28] X.-D. Liu, S. Osher, and T. Chan, Weighted essentially non-oscillatory schemes, Journal of computational physics **115**, 200 (1994).
[29] S. Maneewongvatana and D. M. Mount, Analysis of approximate nearest neighbor searching with clustered point sets, arXiv preprint cs/9901013 (1999).
[30] S. Gottlieb, C.-W. Shu, and E. Tadmor, Strong stability-preserving high-order time discretization methods, SIAM review **43**, 89 (2001).
[31] F. Chollet *et al.*, Keras (2015).
[32] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, TensorFlow: Large-scale machine learning on heterogeneous systems (2015), software available from tensorflow.org.
[33] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).