AGU100 ADVANCING EARTH AND SPACE SCIENCE

Geophysical Research Letters

RESEARCH LETTER
10.1029/2020GL087005

Deep Learning Emulation of Subgrid-Scale Processes in Turbulent Shear Flows

Anikesh Pal[1,2]

[1]National Center for Computational Sciences, Oak Ridge National Laboratory, Oak Ridge, TN, USA, [2]Department of Mechanical Engineering, Indian Institute of Technology Kanpur, Kanpur, India

Key Points:
- DNNs are built to emulate the SGS eddy viscosity and diffusivity for turbulent stratified shear flows
- These DNNs compute the SGS eddy viscosity and diffusivity 2–4 times faster than the dynamic Smagorinsky model
- These DNNs emulate the SGS processes accurately such that the energy and variance budgets match with the dynamic Smagorinsky model

Supporting Information:
- Supporting Information S1

Correspondence to:
A. Pal,
pala@iitk.ac.in

Citation:
Pal, A. (2020). Deep learning emulation of subgrid-scale processes in turbulent shear flows. *Geophysical Research Letters*, *47*, e2020GL087005. https://doi.org/10.1029/2020GL087005

Received 8 JAN 2020
Accepted 13 APR 2020
Accepted article online 23 APR 2020

Abstract Deep neural networks (DNNs) are developed from a data set obtained from the dynamic Smagorinsky model to emulate the subgrid-scale (SGS) viscosity (ν_{sgs}) and diffusivity (κ_{sgs}) for turbulent stratified shear flows encountered in the oceans and the atmosphere. These DNNs predict ν_{sgs} and κ_{sgs} from velocities, strain rates, and density gradients such that the evolution of the kinetic energy budget and density variance budget terms is similar to the corresponding values obtained from the original dynamic Smagorinsky model. These DNNs also compute ν_{sgs} and κ_{sgs} ~2–4 times quicker than the dynamic Smagorinsky model resulting in a ~2–2.5 times acceleration of the entire simulation. This study demonstrates the feasibility of deep learning in emulating the subgrid-scale (SGS) phenomenon in geophysical flows accurately in a cost-effective manner. In a broader perspective, deep learning-based surrogate models can present a promising alternative to the traditional parameterizations of the subgrid-scale processes in climate models.

Plain Language Summary Large eddy simulations (LES) are commonly used to simulate various oceanic and atmospheric flows. In LES, the large eddies are resolved, whereas the small-scale turbulent features, which are the primary sources of mixing, are parameterized using physical models. A deep learning-based surrogate LES model is developed from the data set obtained from such a physical model, the dynamic Smagorinsky model, at moderate Reynolds number and resolution. When this surrogate LES model is deployed for 10 times higher Reynolds number at a relatively higher and lower resolution, it was able to capture all the qualitative and quantitative features of the flow accurately at a cheaper computational cost. The effectiveness of deep learning-based surrogate models to emulate the small-scale processes is a promising area of research and can potentially be extended for various subgrid-scale parameterizations in climate and earth science models.

1. Introduction

The fidelity of climate and earth system models is compromised due to their inability to resolve the turbulent mixing occurring at scales of hundreds of meters to a few kilometers. Stratified oceanic and atmospheric mixing is commonly instigated by a shear instability, which is the dominant mechanism for converting the fluid motion to turbulence. This investigation uses deep learning as a new technique to accurately parameterize the turbulence phenomenon associated with mixing owing to shear instabilities at high Reynolds numbers in a stratified environment.

In recent years, deep learning (Dechter, 1986; LeCun et al., 2015) has evolved as a compelling and cutting-edge topic of research and has demonstrated a tremendous increase in accuracy in the areas of image and speech recognition. Deep learning achieves its great power and flexibility by learning features incrementally through its hidden layer architecture. Owing to its supremacy in terms of accuracy when trained with a huge amount of data, deep neural networks (DNNs) are gaining popularity in the areas of turbulence modeling (Ling et al., 2016; Parish & Duraisamy, 2016; Tracey et al., 2013; Zhang & Duraisamy, 2015), climate and earth science research (Anderson & Lucas, 2018; Bolton & Zanna, 2019; Brenowitz & Bretherton, 2018; Gentine et al., 2018; Pal et al., 2019; Reichstein et al., 2019; Watson, 2019).

A stratified shear layer, characterized by two parallel fluid streams with different velocities and density (see supporting information Figure S1), is commonly observed in the oceans and the atmosphere. Under favorable conditions, such stratified shear layer develops Kelvin-Helmholtz (KH) instabilities and forms large-scale billows that subsequently break down to turbulence and mix the fluid streams. The evolution

of a stratified shear layer is extensively studied by using observational techniques, laboratory experiments, and numerical simulations (Brucker & Sarkar, 2007; Caulfield & Peltier, 2000; Geyer et al., 2010; Pham et al., 2009; Smyth & Moum, 2000a, 2000b). Turbulence-resolving simulations (direct numerical simulations [DNS]) of stratified shear layers have been performed at a Reynolds number of 5,000 by Pham & Sarkar, 2010 and Smyth et al., 2001. However, in a more realistic scenario such as in the oceans and the atmosphere, the Reynolds numbers associated with these shear layers are $O(10^5–10^6)$ (Geyer et al., 2010). DNS at such high Reynolds number are prohibitively expensive. Therefore, large eddy simulations (LES) are used as a cheaper alternative. In LES, the larger eddies are resolved, whereas the effect of the small scales on these large eddies is parameterized by the subgrid-scale fluxes. LES are extensively used in investigating atmospheric boundary layers (Chamecki et al., 2007; Moeng & Sullivan, 1994), shallow cumulus convection (Siebesma et al., 2003), and stratocumulus-topped planetary boundary layer (Stevens et al., 2005). Similarly, LES are also noticeably employed in oceanography (Jalali & Sarkar, 2017; Pham & Sarkar, 2017) and ocean modeling (Chalamalla et al., 2017). LES of the canonical shear layer without stratification has also been performed by Vreman et al. (1997).

Although LES models provide a reasonably accurate representation of the small-scale processes, their implementation in climate models to parameterize turbulence is very challenging. Therefore, a deep learning-based simpler surrogate model is constructed from a robust LES model (dynamic Smagorinsky model, Germano et al., 1991) as a proof of concept to emulate the unresolved turbulent processes accurately. The dynamic Smagorinsky model computes ν_{sgs} and κ_{sgs} from velocities, strain rates, and density gradients. The deep learning-based subgrid-scale model will also use velocities, strain rates, and density gradients to predict ν_{sgs} and κ_{sgs} in a relatively simpler formulation. Machine learning techniques have become popular in recent years for emulating unresolved atmospheric processes. DNNs are used by Gentine et al. (2018) to emulate the effects of unresolved clouds and convection in an idealized simulation over an aqua-planet using the Super-Parameterized Community Atmosphere Model (SPCAM). Rasp et al. (2018) also used DNNs to represent all atmospheric subgrid processes in SPCAM. They have reported close resemblance of mean climate and key aspects of variability between the deep learning-based parameterization and the traditional parameterization in SPCAM. Similar neural network-based parameterization for apparent sources of heat and moisture is developed by Brenowitz and Bretherton (2018) for a near global aqua-planet simulation using the Community Atmosphere Model. A random forest-based parameterization for moist convection is also developed and implemented in a general circulation model (GCM) in an idealized setting by O'Gorman and Dwyer (2018). DNN-based surrogate models are also developed by Pal et al. (2019) for radiative transfer in Super-Parameterized Energy Exascale Earth System Model (SP-E3SM). Bolton and Zanna (2019) trained convolutional neural networks (CNNs) on degraded data from a high-resolution quasigeostrophic ocean model. They demonstrated that their CNNs successfully replicate the spatiotemporal variability of the subgrid eddy momentum forcing, are capable of generalizing to a range of dynamical behaviors, and can be forced to respect global momentum conservation.

In the past few years, neural network-based models have emerged as a substitute to parameterize the subgrid-scale processes in Reynolds-averaged Navier-Stokes (RANS) models (Ling et al., 2016; Parish & Duraisamy, 2016; Tracey et al., 2013; Zhang & Duraisamy, 2015). Recently, Srinivasan et al. (2019) illustrated the potential of neural networks to predict the temporal dynamics of a low-order model of a turbulent flow. However, the use of DNNs to parameterize the subgrid-scale processes in LES is not explored yet. Therefore, the prime motivation of this investigation is to verify the feasibility of deep learning to accurately and efficiently emulate the subgrid-scale processes in LES of geophysical flows. DNNs are developed from a data set obtained from the LES of an evolving shear layer at moderate Reynolds number (10^4) using the dynamic Smagorinsky model. Later, these DNNs replace the dynamic Smagorinsky model to emulate the subgrid-scale processes at a higher Reynolds number (10^5) relevant to the ocean and the atmosphere at different resolutions. The mean and fluctuating statistics obtained from this deep learning-based subgrid-scale model are compared with the traditional dynamic Smagorinsky model to assess the accuracy and effectiveness of this surrogate LES model.

The details of the problem setup and DNN architecture are presented in section 2. Qualitative and quantitative comparisons between the deep learning-based surrogate LES model and the traditional dynamic Smagorinsky model are discussed in section 3, and conclusions are given in section 4.

2. Method

2.1. Problem Setup

2.1.1. Stratified Shear Layer

Following Pham and Sarkar (2014), a stratified shear layer is defined as two parallel streams of fluid with different velocity and density flowing in opposite directions (see Figure S1). The streamwise velocity and density fields are initialized as a function of the vertical direction (x_3) as follows:

$$\langle u_1^* \rangle = \frac{\Delta U_1^*}{2} \tan h \frac{x_3^*}{0.5\delta_{\omega,0}^*}, \tag{1}$$

$$\langle \rho^* \rangle = \rho_0^* + \frac{\Delta \rho^*}{2} \tan h \frac{x_3^*}{0.5\delta_{\omega,0}^*}, \tag{2}$$

where ΔU_1^* and $\Delta \rho^*$ are the initial velocity and density differences, respectively. The initial thickness of the shear layer is $\delta_{\omega,0}^*$ calculated by $\Delta U_1^* / \max(|\frac{d\langle u_1^* \rangle}{dx_3}|)$. The superscript "$*$" denotes dimensional quantities and $\langle \bullet \rangle$ represents averaging in the homogeneous directions (x_1-x_2) as mentioned in equation 2 in supporting information.

2.1.2. Governing Equations and Numerical Method

In a LES model, the equations of motion for an incompressible flow with Boussinesq approximation are filtered in space, nondimensionalized by ΔU_1^*, $\delta_{\omega,0}^*$, and $\Delta \rho^*$ (Pham & Sarkar, 2014), and are written as

mass:

$$\frac{\partial \overline{u}_i}{\partial x_i} = 0, \tag{3}$$

momentum:

$$\frac{\partial \overline{u}_i}{\partial t} + \frac{\partial (\overline{u_j u_i})}{\partial x_j} = -\frac{\partial \overline{p}}{\partial x_i} + \frac{1}{Re_0} \frac{\partial^2 \overline{u}_i}{\partial x_j \partial x_j} + Ri_{b,0} \overline{\rho}' g_i - \frac{\partial \tau_{ij}}{\partial x_j}, \tag{4}$$

density:

$$\frac{\partial \overline{\rho}}{\partial t} + \frac{\partial (\overline{u_j \rho})}{\partial x_j} = \frac{1}{Re_0 Pr_0} \frac{\partial^2 \overline{\rho}}{\partial x_j \partial x_j} - \frac{\partial Q_j}{\partial x_j}, \tag{5}$$

where the overbar denotes the filtered quantities and g is gravity acting in the vertical (x_3) direction. The Reynolds number, $Re_0 = \Delta U_1^* \delta_{\omega,0}^* / \nu^*$ (ν^* is the kinematic viscosity), the bulk Richardson number $Ri_{b,0} = g^* \Delta \rho^* \delta_{\omega,0}^* / \rho_0^* \Delta U_1^{*2}$, and the Prandtl number $Pr_0 = \nu^* / \kappa^*$ (κ^* is the thermal diffusivity) are the three nondimensional parameters that prescribe the fluid motion. Although the Prandtl number $Pr_0 = 1$ in this investigation is typical to the atmosphere, the Reynolds number and the bulk Richardson number considered in this study are well within the range expected in both the ocean and atmosphere. The subgrid-scale stress, τ_{ij}, and subgrid buoyancy flux, Q_j, are parameterized as follows:

$$\tau_{ij} = -2C_d \overline{\Delta}^2 \overline{|S| S_{ij}}, \tag{6}$$

$$Q_j = -C_\theta \overline{\Delta}^2 \overline{|S|} \frac{\partial \overline{\rho}}{\partial x_j}, \tag{7}$$

where $\overline{\Delta}$ is the filter width, C_d is the model coefficient, $\overline{S}_{ij} = 1/2(\partial \overline{u}_i / \partial x_j + \partial \overline{u}_j / \partial x_i)$ is the resolved strain rate, and $\overline{|S|}$ is defined as $\sqrt{2\overline{S}_{ij}\overline{S}_{ij}}$. The subgrid eddy viscosity and diffusivity are given by

$$\nu_{sgs} = C_d \overline{\Delta}^2 \overline{|S|}, \tag{8}$$

$$\kappa_{sgs} = C_\theta \overline{\Delta}^2 \overline{|S|}, \tag{9}$$

respectively. The model coefficients C_d and C_θ are calculated by a dynamic procedure (Germano et al., 1991; Lilly, 1992) in which a test filter is applied to the resolved velocity fields. The quantities denoted by $\widetilde{\bullet}$ are double filtered with both LES and test filters. The dynamic coefficient is computed by

$$C_d = -\frac{1}{2} \frac{\langle L_{ij} M_{ij} \rangle}{\langle M_{ij} M_{ij} \rangle}, \tag{10}$$

where $L_{ij} = \widetilde{\overline{u}_i \overline{u}_j} - \tilde{\overline{u}}_i \tilde{\overline{u}}_j$ and $M_{ij} = \widetilde{\overline{\Delta}^2 |\overline{S}| \overline{S}_{ij}} - \overline{\Delta}^2 \widetilde{|\overline{S}| \overline{S}_{ij}}$. Similarly, the dynamic coefficient for the subgrid buoyancy flux is

$$C_\theta = -\frac{1}{2} \frac{\langle L_i^\theta M_i^\theta \rangle}{\langle M_i^\theta M_i^\theta \rangle}, \tag{11}$$

where $L_i^\theta = \widetilde{\overline{\rho u}_i} - \tilde{\overline{\rho}} \tilde{\overline{u}}_i$ and $M_i = \widetilde{\overline{\Delta}^2 |\overline{S}|} \widetilde{\frac{\partial \overline{\rho}}{\partial x_i}} - \overline{\Delta}^2 \widetilde{|\overline{S}| \frac{\partial \overline{\rho}}{\partial x_i}}$. The ratio of the test and LES filter, $\tilde{\overline{\Delta}}/\overline{\Delta} = 6$, and $\langle \bullet \rangle$ denotes the averaging in the homogeneous directions (in this case, the horizontal, x_1-x_2 direction). For time advancement, a semi-implicit, third-order Runge-Kutta/Crank-Nicolson formulation is used. The viscous terms in the wall normal direction are treated implicitly, whereas the viscous terms in the periodic direction are treated explicitly. All the spatial derivatives are discretized using a central, second-order finite difference scheme on a staggered grid. The pressure Poisson equation, which is utilized to project the velocity field into a divergence-free space, is solved using a multigrid solver. This numerical solver has been validated and used extensively for a number of free-shear and wall-bounded turbulence problems (Brucker & Sarkar, 2010; Pal & Chalamalla, 2020; Pal et al., 2013; Pal & Sarkar, 2015; Pham & Sarkar, 2014, 2018).

The horizontal boundaries have periodic conditions, while the top and bottom boundaries have the following boundary conditions:

$$\overline{u}_1(x_{3,\min}) = \frac{1}{2}, \overline{u}_1(x_{3,\max}) = -\frac{1}{2}, \tag{12}$$

$$\overline{u}_2(x_{3,\min}) = 0, \overline{u}_2(x_{3,\max}) = 0, \tag{13}$$

$$\frac{\partial \overline{u}_3}{\partial x_3}(x_{3,\min}) = \frac{\partial \overline{u}_3}{\partial x_3}(x_{3,\max}) = 0, \tag{14}$$

$$\overline{p}(x_{3,\min}) = \overline{p}(x_{3,\max}) = 0, \tag{15}$$

$$\frac{\partial \overline{\rho}}{\partial x_3}(x_{3,\min}) = \frac{\partial \overline{\rho}}{\partial x_3}(x_{3,\max}) = 0. \tag{16}$$

2.2. DNNs

A dense, fully connected, feed-forward DNN is chosen for this application as follows:

$$h_1 = \varphi_1 \left(W_1 \mathcal{A}_{in} + b_1 \right), \tag{17}$$

$$h_2 = \varphi_2 \left(W_2 h_1 + b_2 \right), \tag{18}$$

$$h_3 = \varphi_3 \left(W_3 h_2 + b_3 \right), \tag{19}$$

$$P_{\text{out}} = \left(W_4 h_3 + b_4 \right). \tag{20}$$

Here, $\mathcal{A}_{in}$ denotes the input vector; P_{out} is the predicted output vector; W_1, W_2, W_3, and W_4 are matrices of trainable weights; b_1, b_2, b_3, and b_4 are "bias vectors"; φ_1, φ_2, and φ_3 are the nonlinear activation functions; and h_1, h_2, and h_3 are the "hidden" vectors whose scalar components are called "neurons." Note that activation functions are only applied on the hidden vectors. For the present investigation, a neural network package called Keras (https://keras.io) has been used, which is a high-level wrapper around TensorFlow (https://www.tensorflow.org/) written in Python.

2.3. Data Collection and Normalization

Table 1 lists all the simulated cases for this study. DSM1 represents simulation at Reynolds number $Re = 10^4$ using the dynamic Smagorinsky model and serves as a source of data collection for a DNN. The domain is decomposed laterally (x_1-x_2 directions) on 480 CPU cores for computation. The input and output variables

Table 1
Simulation Parameters for Shear Layer: N_1, N_2, and N_3 are the Number of Grid Points in x_1, x_2, and x_3 Directions, respectively

Case	N_1	N_2	N_3	Re_0	Δx_1	Δx_2	$\Delta x_{3,\min}$	$\Delta x_{3,\max}$	LES model	SGS CPU time(s)/time step	NS solver CPU time (s)/time step
DSM1 (coarse)	640	192	256	10^4	0.04	0.04	0.04	0.6	*DSM*	1.14	0.276
DSM2 (coarse)	640	192	256	10^5	0.04	0.04	0.04	0.6	*DSM*	1.14	0.276
DLLES2 (coarse)	640	192	256	10^5	0.04	0.04	0.04	0.6	*DNN*	0.42	0.276
DSM3 (coarser)	512	160	256	10^5	0.05	0.05	0.05	0.37	*DSM*	1.02	0.228
DLLES3 (coarser)	512	160	256	10^5	0.05	0.05	0.05	0.37	*DNN*	0.39	0.228
DSM4 (fine)	768	256	512	10^5	0.03	0.03	0.03	0.13	*DSM*	2.94	0.66
DLLES4 (fine)	768	256	512	10^5	0.03	0.03	0.03	0.13	*DNN*	0.81	0.66
DSM5 (finer)	1,024	384	1,024	10^5	0.022	0.022	0.022	0.067	*DSM*	7.71	1.96
DLLES5 (finer)	1,024	384	1,024	10^5	0.022	0.022	0.022	0.067	*DNN*	1.67	1.96

Note. $Re_0 = \frac{\Delta U_1^* \delta_{\omega,0}^*}{\nu^*}$ is the Reynolds number, where $\Delta U_1^* = 1$ m/s is the velocity difference, $\delta_{\omega,0}^* = 1$ m is the initial shear layer thickness, and ν^* (m^2/s) is the kinematic viscosity. The Prandtl number is 1, and the bulk Richardson number is 0.1 for all the cases. The data from DSM1 are used to train the deep learning-based LES model. The size of the computational domain is $L_{x_1}/\delta_{\omega,0}^* = 25.6$, $L_{x_2}/\delta_{\omega,0}^* = 7.6$, and $L_{x_3}/\delta_{\omega,0}^* = 27$ (see Figure S1) in the streamwise, spanwise, and vertical directions, respectively. The grid spacings are nondimesionalized by $\delta_{\omega,0}^*$ and are uniform in streamwise and spanwise directions. In the vertical direction, the grid spacing is kept uniform ($\Delta x_{3,min}$) in the region $-3 < x_3 < 3$. Outside this region, the grids are stretched at a ratio of 0.3–3%.

are collected at each grid point along the vertical column at the center of each CPU core (see the dashed black line in Figure S1) at every 100 time steps. This ensures spatial and temporal variability in the input-output pairs. The dynamic Smagorinsky model computes ν_{sgs} from velocities, strain rates, and the size of the filters (LES and test) at every grid point as equations (8) and (10). Similarly, κ_{sgs} is calculated from velocities, density, strain rates, density gradients in the respective direction, and the size of the filters (LES and test) at every grid point as equations (9) and (11).

The goal is to develop two separate DNNs, one of which will predict ν_{sgs}, whereas the other one will predict κ_{sgs}. The velocities ($\overline{u}_1$, $\overline{u}_2$, and $\overline{u}_3$) and strain rates ($\overline{S}_{ij} = 1/2(\partial\overline{u}_i/\partial x_j + \partial\overline{u}_j/\partial x_i)$ (Germano et al., 1991) at every grid point are taken as inputs to the DNN, whereas ν_{sgs} at the corresponding locations will be the output from the DNN. Analogously, $\overline{u}_1$, $\overline{u}_2$, $\overline{u}_3$, $\overline{\rho}$, $\overline{S}_{ij}$, and $\partial\overline{\rho}/\partial x_i$ will be the inputs for computing κ_{sgs}. The size of the filters is related to the grid size and is taken into account during strain rate calculation. Therefore, the effect of the size of the filters is indirectly associated with the strain rates and is excluded from the inputs to the DNN. Approximately 12 million input-output samples are collected from DSM1.

The following terminology is used to describe the different data sets:

1. Training data set: A random 90% of the saved data from DSM1 is used to train the DNNs over a series of epochs.
2. Testing data set: A random 10% of the saved data from DSM1 is used to probe the accuracy of the DNNs after they have been trained. The DNNs are never trained on this data set during any epoch.
3. Validation data set: During each epoch, a random 10% of the training data set remains unused during a single epoch and is used to track DNN generalization as epochs progress during training.

The input and output variables should be given equal importance while computing the loss function. Therefore, each input and output variable in the present study is normalized by the maximum ($\max_f$) and minimum ($\min_f$) values of each variable across the entire data set using the following relations:

$$\hat{\mathscr{A}}_e = \frac{\mathscr{A}_e - \min_f \mathscr{A}_e}{\max_f \mathscr{A}_e - \min_f \mathscr{A}_e}, \tag{21}$$

$$\hat{\mathscr{C}}_g = \frac{\mathscr{C}_g - \min_f \mathscr{C}_g}{\max_f \mathscr{C}_g - \min_f \mathscr{C}_g}, \tag{22}$$

where $\mathscr{A}_e$ ($e = [1, 9]$) are the inputs ($\overline{u}_1$, $\overline{u}_2$, $\overline{u}_3$, $\overline{S}_{11}$, $\overline{S}_{22}$, $\overline{S}_{33}$, $\overline{S}_{12}$, $\overline{S}_{13}$, and $\overline{S}_{23}$) , $\mathscr{C}_g$ ($g = 1$) is the actual output (ν_{sgs}) from DSM1, and f is the number of training samples. Similarly, $\mathscr{A}_e$ ($e = [1, 13]$) will be $\overline{u}_1$, $\overline{u}_2$, $\overline{u}_3$, $\overline{\rho}$, $\overline{S}_{11}$, $\overline{S}_{22}$, $\overline{S}_{33}$, $\overline{S}_{12}$, $\overline{S}_{13}$, $\overline{S}_{23}$, $\partial\overline{\rho}/\partial x_1$, $\partial\overline{\rho}/\partial x_2$, and $\partial\overline{\rho}/\partial x_3$ for $\mathscr{C}_g$ to be κ_{sgs}. $\hat{\mathscr{A}}_e$ and $\hat{\mathscr{C}}_g$ are used as inputs and output for training the DNNs. Notice that outputs $\hat{\mathscr{C}}_g(\hat{\nu}_{sgs}, \hat{\kappa}_{sgs})$ are normalized (equation (22)) and are

nonnormalized using $\min_f \mathscr{C}_g$ and $\max_f \mathscr{C}_g$ from DSM1 to obtain ν_{sgs} and κ_{sgs}. This is applicable for all the cases with the deep learning-based subgrid-scale model (DLLES2, DLLES3, DLLES4, and DLLES5).

Several exploratory training experiments were carried out to refine the architecture of the DNNs. This includes varying the number of hidden layers, number of neurons, different activation functions, different batch sizes, and different number of epochs. However, the current configuration results in a balance between the validation accuracy and speed-up. The goal was to select a number of hidden layers and neurons per layer that will minimize the training as well as the validation error. These goals were achieved by using three "hidden layers" with 16 neurons in each hidden layer. A ReLU activation unit, which uses the Rectifier Linear (ReL) activation function (https://keras.io/activations/) defined as $f(\mathbf{x}) = \max(0, \mathbf{w} \bullet \mathbf{x})$, is chosen for this study. The optimizer used for training is RMSprop (https://keras.io/optimizers) with a learning rate of 0.01. The DNNs are trained with a batch size of 240 for 200 epochs with *mean squared error* (https://keras.io/losses) as the loss function. The details of the setup and architecture of the DNN such as hidden layers and number of neurons are provided in section 2.2 and Figure S2 in supporting information. Figures S3a and S3b show the evolution of the training and validation loss for ν_{sgs} and κ_{sgs}. The training took approximately 3 hours on a single Nvidia Pascal graphical processing unit (GPU).

Figures S4a and S4b compare the ν_{sgs} and κ_{sgs} predicted from the DNNs with the actual values of ν_{sgs} and κ_{sgs} obtained from the dynamic Smagorinsky model (DSM1) for the first 100 samples of the testing data set. These samples represent the spatial and temporal variability of ν_{sgs} and κ_{sgs} at $Re_0 = 10^4$. The DNNs are able to capture the qualitative and quantitative details of the variation in ν_{sgs} and κ_{sgs} similar to the dynamic Smagorinsky model. The rest of the samples in the testing data set also manifest qualitative and quantitative similarity in ν_{sgs} and κ_{sgs} calculated from the DNNs and DSM1; however, to avoid data clutter in a single plot, only the first 100 samples are shown in Figures S4a and S4b. These two DNNs will now replace the original dynamic Smagorinsky model for shear layer simulations at $Re_0 = 10^5$.

3. Results and Discussion

Figures 1a and 1c compare the distribution of ν_{sgs} between DLLES2 and DSM2 at $Re = 10^5$ at time $t = 51$ and 85 s on a vertical (x_1-x_3) plane at the center ($x_2 = 0$). An analogous contour plot for DLLES4 and DSM4 is shown in Figures 1b and 1d, respectively. It is evident from Figures 1a and 1b that the deep learning-based subgrid-scale model is able to capture the essential dynamical features such as the formation of KH billows and their disintegration to turbulence during the evolution of the shear layer similar to the dynamic Smagorinsky model (Figures 1c and 1d). The DNNs were trained on a data set obtained from a coarse resolution simulation at $Re_0 = 10^4$ (case DSM1). When these DNNs are employed at $Re_0 = 10^5$, relevant to oceanic and atmospheric flows, they predict ν_{sgs} and κ_{sgs} fairly well for same and finer-resolution simulations. This is manifested by the comparison of the vertical profiles of horizontally averaged (see equation 2 in the supporting information) ν_{sgs} at time $t = 51$ and 85 s between the deep learning-based subgrid-scale model and the dynamic Smagorinsky model for the respective cases in Figures 1e and 1f. Analogous comparisons of κ_{sgs} are shown in Figures 1g and 1h. The differences in the vertical profiles of ν_{sgs} and κ_{sgs} between the cases with the deep learning-based subgrid-scale model and the dynamic Smagorinsky model are probably attributed to the difference in Re_0 and grid resolution at which the DNNs are being trained and employed.

The deep learning-based subgrid-scale model accelerates the computation of ν_{sgs} and κ_{sgs} by ~2–4 times as compared to the dynamic Smagorinsky model as evident from the SGS CPU time per time step in Table 1. The computation of ν_{sgs} using the deep learning emulation requires ~0.85 times floating point operations than the dynamic Smagorinky model. The deep learning-based subgrid-scale model replaces all the complex calculations for ν_{sgs} (equations (8) and (10)) and κ_{sgs} (equations (9) and (11)) with matrix-vector multiplications (equations (17)–(20)). An optimized BLAS operation *dgemm* (http://www.netlib.org/lapack/) is used for these matrix multiplications between the input/hidden vectors and the weight matrices. Therefore, the acceleration of the deep learning-based subgrid-scale model is attributed not only to the lesser number of floating point operations but also to the use of highly optimized matrix multipliers designed for modern computer architectures. This speed-up is measured by comparing the minimum and maximum CPU time elapsed in calculating ν_{sgs} and κ_{sgs} among all the CPUs during the entire simulation for the deep learning-based subgrid-scale model and the dynamic Smagorinsky model. The determination of ν_{sgs} and κ_{sgs}, and the Navier-Stokes (NS) solver, which primarily is the Poisson equation (an elliptic equation) for

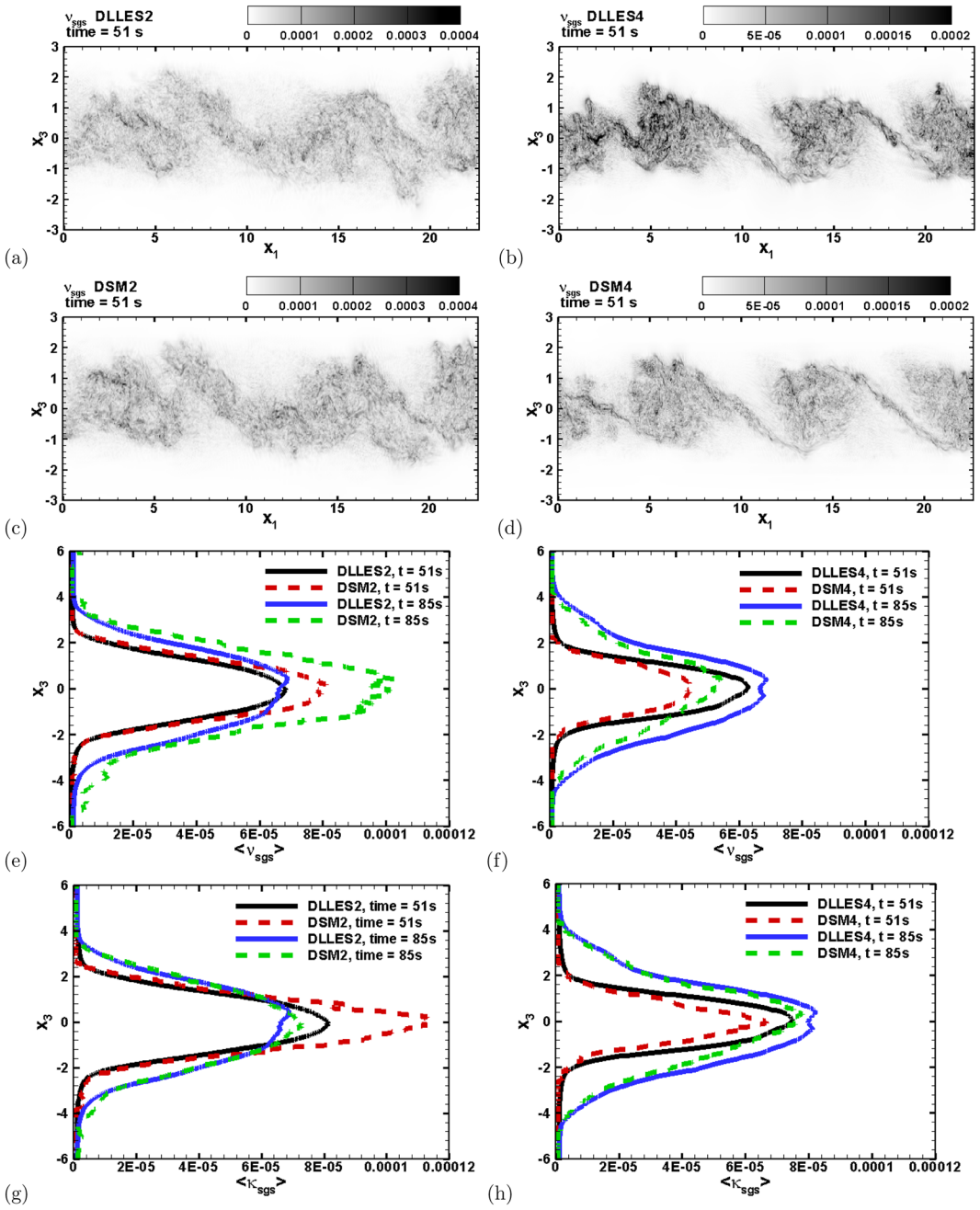

Figure 1. Comparison of ν_{sgs} at central vertical plane ($x_2 = 0$, x_1-x_3 plane) at coarse and fine resolution obtained from deep learning-based LES model (a) DLLES2, (b) DLLES4, and dynamic Smagorinsky model (c) DSM2 and (d) DSM4 at time $t = 51$ s. Comparison of vertical profiles of horizontally averaged $\langle \nu_{sgs} \rangle$ for (e) coarse (DLLES2 and DSM2) and (f) fine-resolution (DLLES4 and DSM4) simulation at time $t = 51{,}85$ s and $Re_0 = 10^5$. Comparison of vertical profiles of horizontally averaged $\langle \kappa_{sgs} \rangle$ for (g) coarse (DLLES2 and DSM2) and (h) fine-resolution (DLLES4 and DSM4) simulation time $t = 51$ and 85 s and $Re_0 = 10^5$.

pressure perturbation, are two of the most computationally expensive modules for the simulation of turbulent shear layers. As indicated in Table 1, the NS solver CPU time is the same for all the respective cases (DSM2 and DLLES2; DSM3 and DLLES3; DSM4 and DLLES4; and DSM5 and DLLES5) and is less than the SGS CPU time. Therefore, an acceleration in the computation of ν_{sgs} and κ_{sgs} by the deep learning-based subgrid-scale model will accelerate the entire simulation by ~2–2.5 times as compared to the simulation using the dynamic Smagorinsky model.

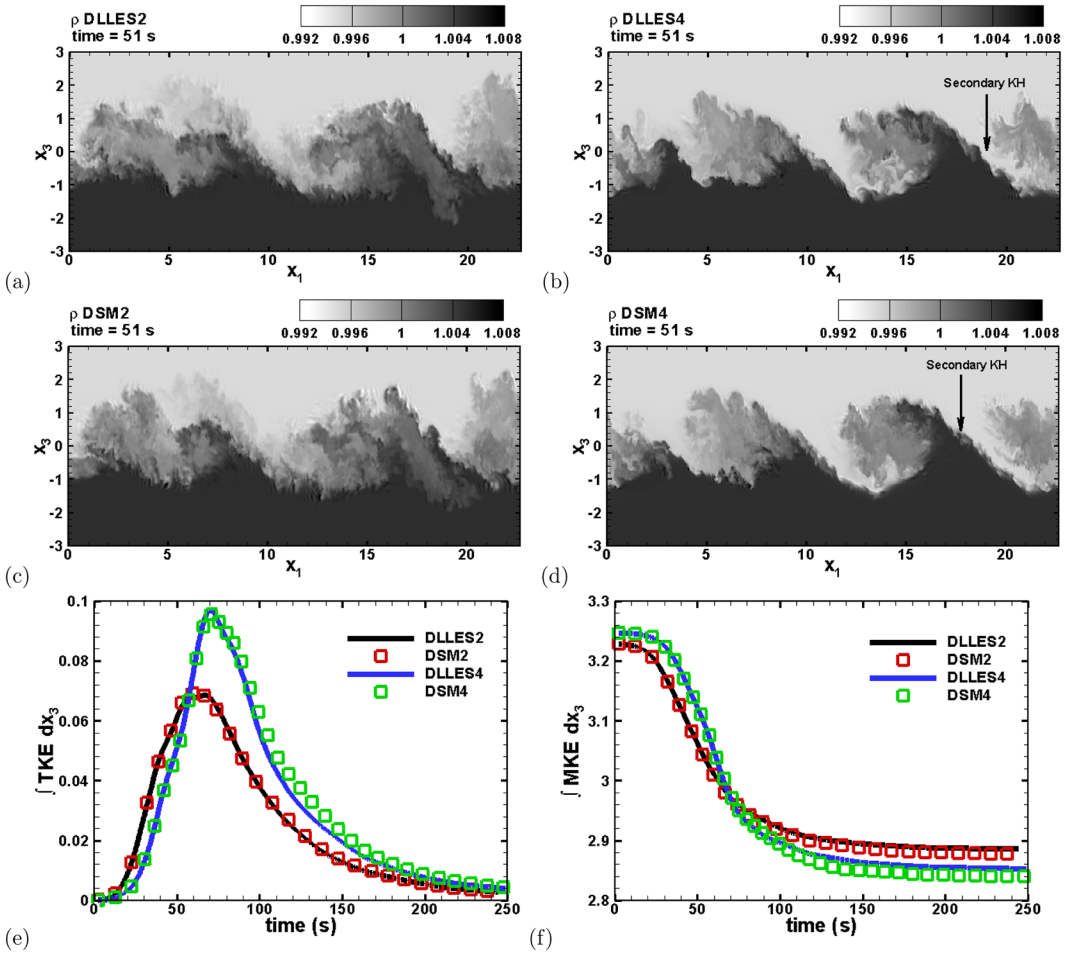

Figure 2. Comparison of filtered density $\overline{\rho}$ field at central vertical plane ($x_2 = 0$, x_1-x_3 plane) at coarse and fine resolution obtained from deep learning-based subgrid-scale model (a) DLLES2, (b) DLLES4, and dynamic Smagorinsky model (c) DSM2 and (d) DSM4 at time $t = 51$ s. Comparison of the time evolution of vertically integrated (e) TKE and (f) MKE between the deep learning-based subgrid-scale model and the dynamic Smagorinsky model at $Re_0 = 10^5$.

The DNNs developed in this investigation have learned the underlying physics of the problem resulting in their capability to establish a relationship between the flow variables and eddy viscosity/diffusivity. This is the reason why these DNNs are able to predict ν_{sgs} and κ_{sgs} for a higher Re_0 at different resolutions fairly well. It should be noted that a flow with a different physics will require an additional simulation for data collection and development of a deep learning-based SGS model for that particular flow. However, as demonstrated in this study, the deep learning-based SGS model can not only provide accuracy but will overall be computationally cheaper (see the supporting information for total computational time details) when used for parametric studies as compared to the dynamic Smagorinsky model.

A qualitatively similar evolution of the filtered density field ($\overline{\rho}$) is observed between the deep learning-based subgrid-scale model and the dynamic Smagorinsky model for the coarse (Figures 2a and 2c) and fine (Figures 2b and 2d) grid simulations. An analogous comparison for the coarser (DLLES3 and DSM3) and finer (DLLES5 and DSM5) grid simulations are shown in Figures S5a,c and S5b,d respectively. The deep learning-based subgrid-scale model is able to capture the details of the flow field such as the secondary KH instabilities along the braids of the primary KH billows similar to the dynamic Smagorinsky model. The appearance of such secondary KH instabilities is typical to high Re_0 shear flows (Geyer et al., 2010). It can also be observed from Figures 2e and 2f that the deep learning-based subgrid-scale model imitates the time evolution of the vertically integrated turbulent kinetic energy (*TKE*) and mean kinetic energy (*MKE*) similar to the dynamic Smagorinsky model for the coarse (DLLES2 and DSM2) and fine (DLLES4 and DSM4) grid simulations. The increase in *TKE* is attributed to the decrease in *MKE* signifying the conversion of *MKE* to *TKE*. This phenomenon is further explained by means of the production term in *TKE* budget equation

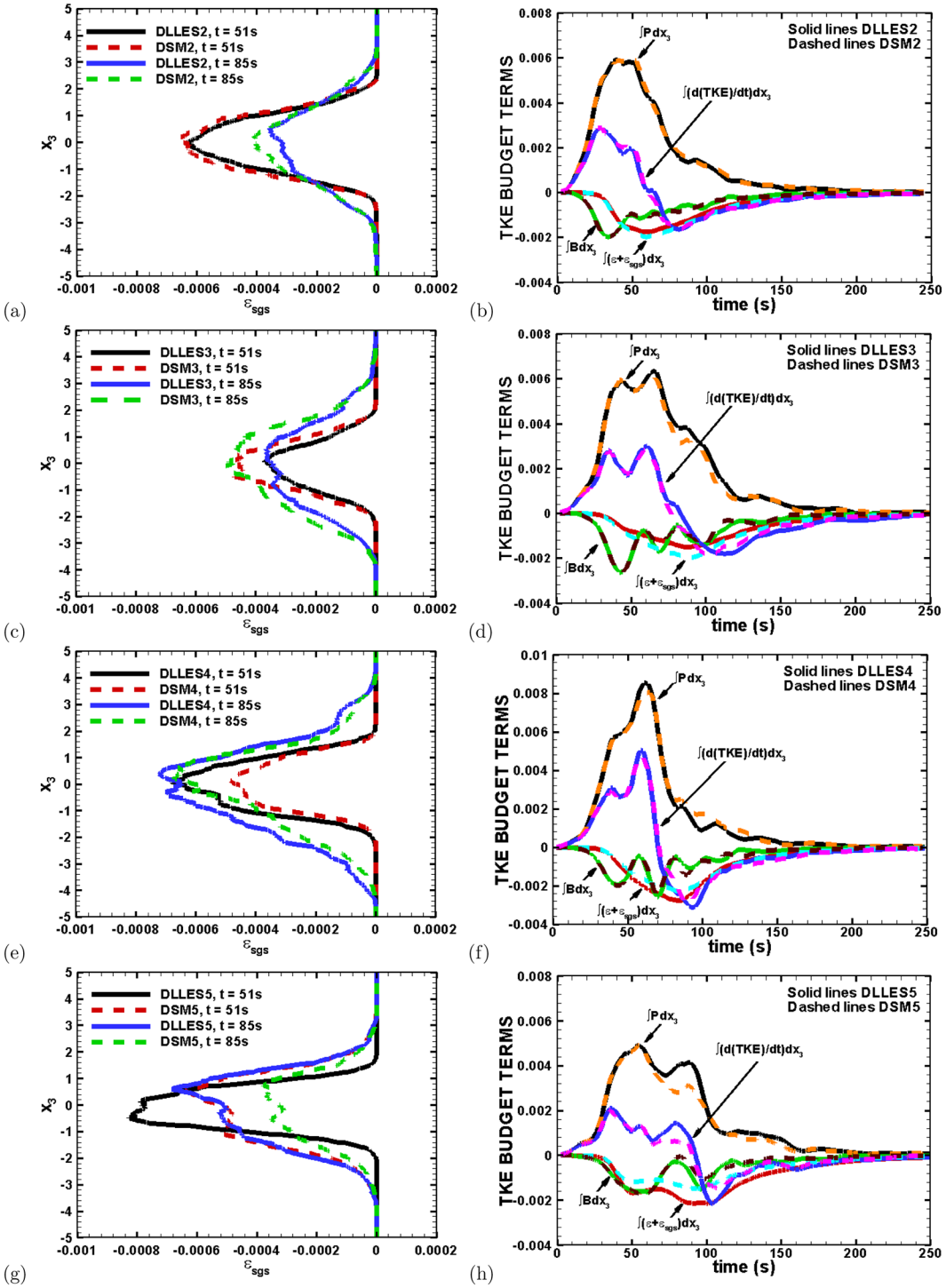

Figure 3. Comparison of vertical profiles ε_{sgs} between (a) DLLES2 and DSM2, (c) DLLES3 and DSM3, (e) DLLES4 and DSM4, and (g) DLLES5 and DSM5 at time $t = 51$ and 85 s and $Re_0 = 10^5$. Comparison of vertically integrated TKE budget terms between (b) DLLES2 and DSM2, (d) DLLES3 and DSM3, (f) DLLES4 and DSM4, and (h) DLLES5 and DSM5 at $Re_0 = 10^5$.

in the upcoming paragraphs. Figures S5e and S5f also corroborate the accuracy of the deep learning-based subgrid-scale model in computing *TKE* and *MKE* for coarser (DLLES3) and finer (DLLES5) grid simulations.

In LES, the large scales are resolved, whereas the small-scale turbulence is modeled using a subgrid-scale model. The performance of a subgrid-scale model is determined by its ability to provide sufficient subgrid-scale dissipation and scalar dissipation via ν_{sgs} and κ_{sgs}, respectively. If the subgrid-scale/scalar dissipation is low, energy builds up in small scales resulting in eventual crashing of the solver. If the subgrid-scale/scalar dissipation is high, the small-scale features dissipate resulting in lower turbulence intensities and disappearance of the finer structures in large scales resulting in inaccurate evolution of the large structures. Therefore, to verify the accuracy of the deep learning-based subgrid-scale model, it is imperative to compare the small-scale (*TKE*, scalar variance budgets) and large-scale statistics (*MKE* budget) obtained from the deep learning algorithm and the dynamic Sagorinsky model. The *TKE* budget (equations (3)–(8)), density variance budget (equations (13)–(18)), and *MKE* budget (equations (9)–(12)) terms are presented in the supporting information.

The vertical profiles of subgrid dissipation (ε_{sgs}, equation (7) in the supporting information) at time $t = 51$ and 85 s for different grid resolutions are shown in Figures 3a, 3c, 3e and 3g. The resolved dissipation (ε, equation (5) in the supporting information) is small as compared to the subgrid dissipation for all the cases and is not shown. This is attributed to high Re_0 at which subgrid dissipation contributes more to the *TKE* budget, making up for the smaller-resolved dissipation. Pham and Sarkar (2014) presented LES studies at $Re_0 = 5{,}000$ in which both the resolved and subgrid-scale dissipation contributes equally. The deep learning-based LES model is able to determine the subgrid-scale dissipation at time $t = 51$ and 85 s fairly well for all the grid resolutions when compared with the dynamic Smagorinksy model. The differences in the vertical profiles of ε_{sgs} is directly related to the differences in the vertical profiles of ν_{sgs}. Geyer et al. (2010) reported that the KH instabilities at $Re_0 = 1.25 \times 10^5$, $Ri_b \sim 0.2$–0.25, and $Pr_0 = 700$ has dissipation in the range of 3–6.8×10^{-4} m^2 s^{-3}. Note that Geyer et al. (2010) have defined Reynolds number based on half the velocity difference across the layer and halfwidth of the shear zone. Therefore, their Reynolds number of 5×10^5 is equal to $Re_0 = 1.25 \times 10^5$. The dissipation from the deep learning-based LES model is in the range 3.6–7×10^{-4} m^2 s^{-3} (Figures 3a, 3c, 3e and 3g) similar to the observations of Geyer et al. (2010).

A comparison of the time evolution of the vertically integrated *TKE* budget terms between the deep learning-based subgrid-scale model and the dynamic Smagorinsky model for different grid resolutions is shown in Figures 3b, 3d, 3f and 3h. The resolved and subgrid transport terms when integrated over the computational domain are negligible and are not shown. The initial increase in the production term represents the extraction of energy from the shear, as also manifested by the decrease in *MKE* in Figure 2f and Figure S5f. This energy is converted to turbulence as shown by the increase in $d(TKE)/dt$ term. At a later time ($t > 60$ s), the turbulence decays owing to the loss to dissipation and buoyancy flux. The differences in *TKE* budget terms among different cases are attributed to the different grid resolutions. The deep learning-based subgrid-scale model is able to mimic the magnitude and variation of the production (P), buoyancy flux (B), and the advection ($d(TKE)/dt$) terms at different grid resolutions similar to the dynamic Smagorinsky model. There are small differences in the evolution of the total dissipation ($\varepsilon + \varepsilon_{sgs}$), which is attributed to the differences in ν_{sgs} computed by the DNNs amd the dynamic Smagorinsky model.

Comparison of the time evolution of vertically integrated *MKE* budget terms (Figures 4a, 4c, 4e and 4g) and density variance budget terms (Figures 4b, 4d, 4f and 4h) between the deep learning-based subgrid-scale model and the dynamic Smagorinsky model for the cases at $Re_0 = 10^5$ is shown in Figure 4. The purpose of analyzing the *MKE* budget is to appraise the performance of the deep learning-based subgrid-scale model in the conservation of the mean kinetic energy. The *MKE* transport terms, *MKE* dissipation terms, and mean buoyancy production term (see equations (10)–(12) in the supporting information) become negligible when integrated in the vertical direction. Therefore, production is the only source of the rate of change of *MKE*. The deep learning-based subgrid-scale model respects the conservation of *MKE* for all grid resolutions similar to the dynamic Smagorinky model as shown in Figures 4a, 4c, 4e and 4g. The scalar transport terms (resolved and subgrid) when integrated vertically also become negligible and are therefore not shown in Figures 4b, 4d, 4f and 4h. The scalar production (P_ρ) term is the source, whereas the scalar dissipation ($\chi_\rho + \chi_{\rho,sgs}$) acts as a sink for the evolution of scalar variance advection term ($\frac{d\langle\tilde{\rho}^{''2}\rangle}{dt}$). Once again, the deep learning-based subgrid-scale model successfully captures the variation of the scalar production and scalar variance advection

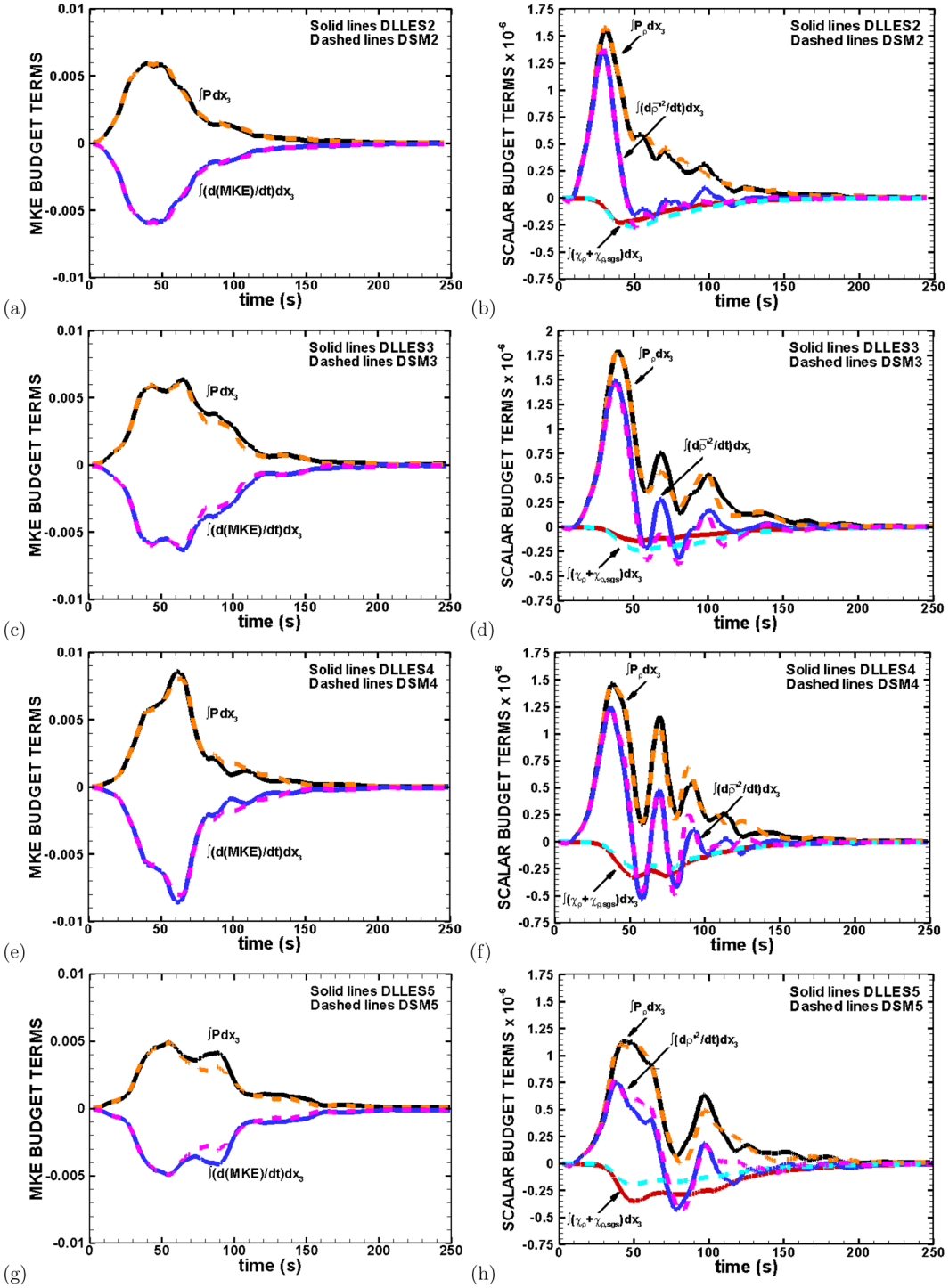

Figure 4. Comparison of vertically integrated *MKE* budget terms (a) DLLES2 and DSM2, (c) DLLES3 and DSM3, (e) DLLES4 and DSM4, and (g) DLLES5 and DSM5. Comparison of vertically integrated density variance budget terms between (b) DLLES2 and DSM2, (d) DLLES3 and DSM3, (f) DLLES4 and DSM4, and (h) DLLES5 and DSM5 at $Re_0 = 10^5$.

terms similar to the dynamic Smagorinsky model for all the grid resolutions at $Re_0 = 10^5$ with small differences in the evolution of the scalar dissipation term. These differences are associated with the differences in κ_{sgs} calculated from the deep learning-based subgrid-scale model and the dynamic Smagorinsky model.

4. Conclusions

Deep learning is used as a new technique to emulate the subgrid-scale viscosity and diffusivity for LES of turbulent shear flows at Reynolds number relevant to that observed in the oceans and the atmosphere. The deep learning-based subgrid-scale model is developed from a data set obtained by simulating a shear layer using the traditional dynamic Smagorinsky model at a particular grid resolution at $Re_0 = 10^4$. Once this subgrid-scale model is trained and validated, it replaces the dynamic Smagorinsky model for simulating shear flow at $Re_0 = 10^5$ for different grid configurations. The deep learning-based subgrid-scale model predicts ν_{sgs} and κ_{sgs} from velocities, strain rates, and density gradients such that all the qualitative features during the evolution of the shear layer such as the KH billows, their instabilities, secondary KH instabilities in the braids, and eventual transition to turbulence are captured for not only the same grid resolution but for a finer and coarser grid resolution as well. This model also computes the subgrid-scale viscosity and diffusivity ∼2–4 times quicker than the dynamic Smagorinsky model. This acceleration of the deep learning-based subgrid-scale model is primarily attributed to the simplification of the complex computation of the subgrid-scale viscosity and diffusivity to vector matrix multiplications. As the computational time required to determine ν_{sgs} and κ_{sgs} turns out to be the prime computational burden for the present cases, a ∼2–4 times acceleration by the deep learning-based subgrid-scale model results in a ∼2–2.5 times speed-up of the entire simulation. A deep learning-based subgrid-scale model thus proves to be computationally efficient when used for parametric studies. The development of deep learning-based subgrid-scale models may require data from additional simulations. However, the consolidated cost of the additional simulations and training the DNNs will be compensated as the deep learning-based subgrid-scale model accelerates the cases with different parameters. Quantitative analysis reveals that the deep learning-based subgrid-scale model conserves mean kinetic energy during the evolution of the shear layer similar to the dynamic Smagorinsky model. This surrogate LES model is also able to mimic the terms of the turbulent kinetic energy budget and the density variance budget as the dynamic Smagorinsky model.

The primary motivation of developing a deep learning-based subgrid-scale model is to verify the feasibility of machine learning algorithms to determine the subgrid-scale processes accurately in large eddy simulations of geophysical flows. This study builds a confidence in experimenting with deep learning alogrithms in emulating the subgrid-scale processes in climate models similar to Gentine et al. (2018) and Rasp et al. (2018). The subgrid-scale processes such as rain, clouds, and mixing owing to turbulence are typically represented by physical models in current climate models. These physical models are sometimes complicated and introduce uncertainties in the outcomes of low-resolution climate simulations. The deep learning-based subgrid-scale models developed from a high-resolution climate simulation data set will be a simpler representation of the subgrid-scale processes and can easily be implemented in the climate model. Although this investigation was carried out as a proof of concept, the results are promising enough to motivate futuristic research in data-driven emulation of the small-scale processes in climate and earth science models. The capability of DNNs in parameterizing the turbulent processes at finer as well as coarser resolutions accurately at reduced computation cost has been demonstrated in this study. Therefore, DNN-based SGS parameterization can be an alternative to the SGS parameterizations based on physically motivated approaches in ocean (Jansen et al., 2015) and climate (Pieroth et al., 2018) models. Attempts similar to Gentine et al. (2018) and Pal et al. (2019) will be made in the future to implement deep learning-based subgrid-scale models to emulate the different subgrid-scale processes in a climate model.

Data and Materials Availability

The code for training the DNN is available at https://doi.org/10.5281/zenodo.3748123. This repository also contains the weights matrices and bias vectors. The data set for DNN training and testing amount to several GB is available from the author upon request.

Acknowledgments
This research used the resources of the Oak Ridge Leadership Computing Facility, which is a Department of Energy Office of Science User Facility supported under Contract DE-AC05-00OR22725.

References

Anderson, G. J., & Lucas, D. D. (2018). Machine learning predictions of a multiresolution climate model ensemble. *Geophysical Research Letters*, *45*, 4273–4280. https://doi.org/10.1029/2018GL077049

Bolton, T., & Zanna, L. (2019). Applications of deep learning to ocean data inference and subgrid parameterization. *Journal of Advances in Modeling Earth Systems*, *11*, 376–399. https://doi.org/10.1029/2018MS001472

Brenowitz, N. D., & Bretherton, C. S. (2018). Prognostic validation of a neural network unified physics parameterization. *Geophysical Research Letters*, *45*, 6289–6298. https://doi.org/10.1029/2018GL078510

Brucker, K. A., & Sarkar, S. (2007). Evolution of an initially turbulent stratified shear layer. *Physics of Fluids*, *19*(10), 105105.

Brucker, K. A, & Sarkar, S (2010). A comparative study of self-propelled and towed wakes in a stratified fluid. *Journal of Fluid Mechanics*, *652*, 373–404.

Caulfield, C., & Peltier, W. (2000). The anatomy of the mixing transition in homogeneous and stratified free shear layers. *Journal of Fluid Mechanics*, *413*, 1–47.

Chalamalla, V. K., Santilli, E., Scotti, A., Jalali, M., & Sarkar, S. (2017). SOMAR-LES: A framework for multi-scale modeling of turbulent stratified oceanic flows. *Ocean Modelling*, *120*, 101–119.

Chamecki, M., Meneveau, C., & Parlange, M. B. (2007). The local structure of atmospheric turbulence and its effect on the Smagorinsky model for large eddy simulation. *Journal of the Atmospheric Sciences*, *64*(6), 1941–1958.

Dechter, R (1986). *Learning while searching in constraint-satisfaction problems*.

Gentine, P., Pritchard, M., Rasp, S., Reinaudi, G., & Yacalis, G. (2018). Could machine learning break the convection parameterization deadlock? *Geophysical Research Letters*, *45*, 5742–5751. https://doi.org/10.1029/2018GL078202

Germano, M., Piomelli, U., Moin, P., & Cabot, W. H. (1991). A dynamic subgrid-scale eddy viscosity model. *Physics of Fluids A: Fluid Dynamics*, *3*(7), 1760–1765.

Geyer, W. R., Lavery, A. C., Scully, M. E., & Trowbridge, J. H. (2010). Mixing by shear instability at high Reynolds number. *Geophysical Research Letters*, *37*, L22607. https://doi.org/10.1029/2010GL045272

Jalali, M., & Sarkar, S. (2017). Large eddy simulation of flow and turbulence at the steep topography of Luzon Strait. *Geophysical Research Letters*, *44*, 9440–9448. https://doi.org/10.1002/2017GL074119

Jansen, M. F., Held, I. M., Adcroft, A., & Hallberg, R. (2015). Energy budget-based backscatter in an eddy permitting primitive equation model. *Ocean Modelling*, *94*, 15–26.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436.

Lilly, D. K. (1992). A proposed modification of the Germano subgrid-scale closure method. *Physics of Fluids A: Fluid Dynamics*, *4*(3), 633–635.

Ling, J., Kurzawski, A., & Templeton, J. (2016). Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *Journal of Fluid Mechanics*, *807*, 155–166.

Moeng, C.-H., & Sullivan, P. P. (1994). A comparison of shear-and buoyancy-driven planetary boundary layer flows. *Journal of the Atmospheric Sciences*, *51*(7), 999–1022.

O'Gorman, P. A., & Dwyer, J. G. (2018). Using machine learning to parameterize moist convection: Potential for modeling of climate, climate change, and extreme events. *Journal of Advances in Modeling Earth Systems*, *10*, 2548–2563. https://doi.org/10.1029/2018MS001351

Pal, A., & Chalamalla, V. K. (2020). Evolution of plumes and turbulent dynamics in deep-ocean convection. *Journal of Fluid Mechanics*, *889*, A35.

Pal, A., de Stadler, M. B., & Sarkar, S. (2013). The spatial evolution of fluctuations in a self-propelled wake compared to a patch of turbulence. *Physics of Fluids*, *25*(9), 95106.

Pal, A., Mahajan, S., & Norman, M. R. (2019). Using deep neural networks as cost-effective surrogate models for super-parameterized E3SM radiative transfer. *Geophysical Research Letters*, *46*, 6069–6079. https://doi.org/10.1029/2018GL081646

Pal, A., & Sarkar, S. (2015). Effect of external turbulence on the evolution of a wake in stratified and unstratified environments. *Journal of Fluid Mechanics*, *772*, 361–385.

Parish, E. J., & Duraisamy, K (2016). A paradigm for data-driven predictive modeling using field inversion and machine learning. *Journal of Computational Physics*, *305*, 758–774.

Pham, H. T., & Sarkar, S. (2010). Transport and mixing of density in a continuously stratified shear layer. *Journal Turbulence*, *11*, N24.

Pham, H. T, & Sarkar, S (2014). Large eddy simulations of a stratified shear layer. *Journal of Fluids Engineering*, *136*(6), 60,913.

Pham, H. T., & Sarkar, S. (2017). Turbulent entrainment in a strongly stratified barrier layer. *Journal of Geophysical Research: Oceans*, *122*, 5075–5087. https://doi.org/10.1002/2016JC012357

Pham, H. T., & Sarkar, S. (2018). Ageostrophic secondary circulation at a submesoscale front and the formation of gravity currents. *Journal of Physical Oceanography*, *48*, 2507–2529.

Pham, H. T., Sarkar, S, & Brucker, K. A (2009). Dynamics of a stratified shear layer above a region of uniform stratification. *Journal of Fluid Mechanics*, *630*, 191–223.

Pieroth, M., Dolaptchiev, S. I., Zacharuk, M., Heppelmann, T., Gritsun, A., & Achatz, U. (2018). Climate dependence in empirical parameters of subgrid-scale parameterizations using the fluctuation–dissipation theorem. *Journal of the Atmospheric Sciences*, *75*(11), 3843–3860.

Rasp, S., Pritchard, M. S., & Gentine, P. (2018). Deep learning to represent subgrid processes in climate models. *Proceedings of the National Academy of Sciences*, *115*(39), 9684–9689.

Reichstein, M., Camps-Valls, G., Stevens, B., Jung, M., Denzler, J., Carvalhais, N., & Prabhat (2019). Deep learning and process understanding for data-driven earth system science. *Nature*, *566*(7743), 195.

Siebesma, A. P., Bretherton, C. S., Brown, A., Chlond, A., Cuxart, J., Duynkerke, P. G., et al. (2003). A large eddy simulation intercomparison study of shallow cumulus convection. *Journal of the Atmospheric Sciences*, *60*(10), 1201–1219.

Smyth, W. D., & Moum, J. N. (2000a). Anisotropy of turbulence in stably stratified mixing layers. *Physics of Fluids*, *12*(6), 1343–1362.

Smyth, W. D., & Moum, J. N. (2000b). Length scales of turbulence in stably stratified mixing layers. *Physics of Fluids*, *12*(6), 1327–1342.

Smyth, W., Moum, J., & Caldwell, D. (2001). The efficiency of mixing in turbulent patches: Inferences from direct simulations and microstructure observations. *Journal of Physical Oceanography*, *31*(8), 1969–1992.

Srinivasan, P., Guastoni, L, Azizpour, H, Schlatter, P, & Vinuesa, R (2019). Predictions of turbulent shear flows using deep neural networks. *Physical Review Fluids*, *4*(5), 54,603.

Stevens, B., Moeng, C.-H., Ackerman, A. S., Bretherton, C. S., Chlond, A., de Roode, S., et al. (2005). Evaluation of large-eddy simulations via observations of nocturnal marine stratocumulus. *Monthly Weather Review*, *133*(6), 1443–1462.

Tracey, B., Duraisamy, K. K., & Alonso, J. (2013). Application of supervised learning to quantify uncertainties in turbulence and combustion modeling. In *51st AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition* (pp. 259).

Vreman, B., Geurts, B., & Kuerten, H. (1997). Large-eddy simulation of the turbulent mixing layer. *Journal of Fluid Mechanics*, *339*, 357–390.
Watson, P. A. (2019). Applying machine learning to improve simulations of a chaotic dynamical system using empirical error correction. *Journal of Advances in Modeling Earth Systems*, *11*, 1402–1417. https://doi.org/10.1029/2018MS001597
Zhang, Z. J., & Duraisamy, K. (2015). Machine learning methods for data-driven turbulence modeling. In *22nd AIAA Computational Fluid Dynamics Conference* (pp. 2460).