Electronic Transactions on Numerical Analysis.
Volume 56, pp. 117–137, 2022.
Copyright © 2022, Kent State University.
ISSN 1068–9613.
DOI: 10.1553/etna_vol56s117

ETNA
Kent State University **and**
Johann Radon Institute (RICAM)

A MACHINE LEARNING FRAMEWORK FOR LES CLOSURE TERMS*

MARIUS KURZ† AND ANDREA BECK‡

Abstract. In the present work, we explore the capability of artificial neural networks (ANN) to predict the closure terms for large eddy simulations (LES) solely from coarse-scale data. To this end, we derive a consistent framework for LES closure models, with special emphasis laid upon the incorporation of implicit discretization-based filters and numerical approximation errors. We investigate implicit filter types that are inspired by the solution representation of discontinuous Galerkin and finite volume schemes and mimic the behavior of the discretization operator, and a global Fourier cutoff filter as a representative of a typical explicit LES filter. Within the perfect LES framework, we compute the exact closure terms for the different LES filter functions from direct numerical simulation results of decaying homogeneous isotropic turbulence. Multiple ANN with a multilayer perceptron (MLP) or a gated recurrent unit (GRU) architecture are trained to predict the computed closure terms solely from coarse-scale input data. For the given application, the GRU architecture clearly outperforms the MLP networks in terms of accuracy, whilst reaching up to 99.9% correlation between the networks' predictions and the exact closure terms for all considered filter functions. The GRU networks are also shown to generalize well across different LES filters and resolutions. The present study can thus be seen as a starting point for the investigation of data-based modeling approaches for LES, which not only include the physical closure terms, but account for the discretization effects in implicitly filtered LES as well.

Key words. large eddy simulation, turbulence models, deep learning, artificial neural networks, recurrent neural networks

AMS subject classifications. 76F65, 68T07, 76F05

1. Introduction.

Over the last decade, machine learning methods, and in particular artificial neural networks (ANN), have achieved tremendous success: from pushing the state-of-the-art in the fields of image and speech recognition [18, 19] to surpassing human-level performance in the game of Go [31, 32]. This recent success of ANN was predominantly fueled by the emergence of large datasets, the exploitation of highly-parallel graphical processing units (GPU) [18], and the development of easy-to-use high-performance machine learning libraries like PyTorch [26] and TensorFlow [1]. In general, artificial neural networks can approximate any continuous functional relationship between input and output quantities solely based on data and without prior assumptions on the nature of said function. These approximation properties are shown by a variety of universal approximation theorems in the literature, e.g., [7, 15, 21].

Significant efforts have been directed towards utilizing the approximation capabilities of neural networks also for problems in other scientific fields, including the field of turbulence research. Turbulence is a multi-scale phenomenon, for which the range of active scales in the flow grows with increasing Reynolds numbers. Thus, direct numerical simulation (DNS) at high Reynolds numbers is still computationally prohibitive for most applications. Common approaches to mitigate the computational effort are the Reynolds-averaged Navier-Stokes equations (RANS) or the method of large eddy simulation (LES), which both rely on filter operations to resolve only the most influential part of the solution, while modeling the effects of the non-resolved part. The filtering of the non-linear convective terms of the Navier-Stokes equations evokes the so-called closure problem, which introduces an additional model term in the coarse-scale equations. For the RANS approach, only the temporally averaged solution is resolved and the influence of the turbulent fluctuations is modeled, while the LES approach

*Received September 30, 2020. Accepted October 6, 2021. Published online on January 10, 2022. Recommended by Xiao-Chuan Cai.

†Institute of Aerodynamics and Gas Dynamics, University of Stuttgart, Pfaffenwaldring 21, 70569 Stuttgart, Germany (`marius.kurz@iag.uni-stuttgart.de`).

‡Laboratory of Fluid Dynamics and Technical Flows, University of Magdeburg "Otto von Guericke", Universitätsplatz 2, 39106 Magdeburg, Germany (`andrea@beck.aero`).

resolves the large, energy-containing flow scales in space and time, but uses models to describe the non-resolved fine scale contributions. These models are typically derived based on physical or mathematical considerations. Despite decades of research, no model proved itself superior for all applications and many models comprise empirical parameters, which have to be tuned for the respective flow regime and/or the discretization choices [9].

Recently, increasing efforts have been made to complement the established physics-based methodology of turbulence modeling by a data-driven approach based on ANN. In one of the first works employing neural networks for LES modeling, Sarghini et al. [30] used a neural network to approximate a mixed LES model to save computation time. A similar approach for RANS was investigated in [35]. Ling et al. [20] derived a novel ANN architecture to embed physical invariances in the predicted RANS closure terms. In context of LES, Maulik and San [22] used ANN for approximate deconvolution of filtered turbulence. The direct inference of the LES subgrid stresses is investigated in [23] for two-dimensional Kraichnan turbulence and in [8] for turbulent channel flow. Different network architectures were used by Beck et al. in [3], who used convolutional neural networks to predict the closure terms for homogeneous isotropic turbulence (HIT), and Srinivasan et al. [34], who used long short-term memory networks to predict the turbulent dynamics by means of a shear flow model. In [36], ANN are used to predict the closure for compressible isotropic turbulence. More recently, Novati et al. [25] used a multi-agent reinforcement learning approach to infer local LES model parameters for the HIT test case.

In this work, we derive a general machine learning framework for LES closure models, while laying emphasis on discretization effects and the influence of different LES filter forms. We generate training data by applying several distinct LES filter functions to DNS data of decaying homogeneous isotropic turbulence (DHIT). For this, we project the DNS solution onto a discontinuous Galerkin (DG) or finite volume (FV) flavored representation on a coarsened mesh. This resembles the implicitly filtered LES approach, where the discretization acts as an implicit LES filter. In addition, a global Fourier cutoff filter is used, while ensuring that the filtered solution is perfectly resolved by the underlying numerical scheme to mitigate any influences of the LES discretization. We show that the multilayer perceptron (MLP) and recurrent neural networks with gated recurrent units (GRU) are generally able to learn the unknown closure terms from data for all filter forms, with the GRU achieving excellent accuracy with up to 99.9% correlation between the predicted and the exact closure terms. Further, we show that the trained GRU networks generalize well across different LES resolutions and different filter functions. This is a surprising and encouraging result, since an accurate prediction of the actual closure terms would greatly benefit the development and help guide the selection of proper LES closure models.

This paper is organized as follows: In Section 2, the underlying equations are discussed and special emphasis is laid upon the derivation of the perfect LES closure terms. The relevant network architectures are discussed in Section 3 and the training of the networks is described in Section 4. The results are reported in Section 5. Section 6 concludes the paper.

2. Turbulence modeling.

Since neural networks approximate functional relationships solely from data, special care has to be taken to ensure its validity and consistency. Based on the compressible Navier-Stokes equations in Section 2.1, the LES closure terms are derived in a consistent manner in Section 2.2 as the overall target quantity of the machine learning task. For the DHIT test case described in Section 2.3, these closure terms are then computed and examined in Section 2.4.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

2.1. Governing equations. The Navier-Stokes equations describe the evolution of compressible, viscous fluids and can be written in conservative form as

$$U_t + \nabla_x \cdot F^c(U) - \nabla_x \cdot F^v\left(U, \nabla_x U\right) = 0. \tag{2.1}$$

The vector of conserved variables is $U = [\rho, \rho v_1, \rho v_2, \rho v_3, \rho e]^T$ with mass, momentum, and energy, respectively, U_t denotes differentiation with respect to time, and the differential operator with respect to the spatial coordinates is denoted as ∇_x. The convective fluxes F^c and the viscous fluxes F^v with columns $i = 1, 2, 3$ take the form

$$F_i^c = \begin{bmatrix} \rho v_i \\ \rho v_1 v_i + \delta_{1i} p \\ \rho v_2 v_i + \delta_{2i} p \\ \rho v_3 v_i + \delta_{3i} p \\ \rho e v_i + p v_i \end{bmatrix}, \; F_i^v = \begin{bmatrix} 0 \\ \tau_{1i} \\ \tau_{2i} \\ \tau_{3i} \\ \tau_{ij} v_j - q_i \end{bmatrix},$$

with δ denoting the Kronecker delta and p the static pressure. The stress tensor τ_{ij} and the heat flux q_i can be written as

$$\begin{aligned} \tau_{ij} &= \mu \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} - \tfrac{2}{3} \delta_{ij} \frac{\partial v_k}{\partial x_k} \right), \\ q_i &= -k \, \frac{\partial T}{\partial x_i}, \end{aligned} \tag{2.2}$$

where k denotes the heat conductivity, T is the static temperature, and μ is the dynamic viscosity. Assuming an ideal gas, the equation system is closed by the relation

$$p = \rho\left(\gamma - 1\right) \left[e - \tfrac{1}{2}\left(v_1^2 + v_2^2 + v_3^2\right) \right],$$

with γ as ratio of specific heats. In this work, the compressible equations are only solved for small Mach numbers, i.e., in the limit $\mathrm{Ma} \to 0$, for which compressibility effects become negligible and eventually the incompressible Navier-Stokes equations are solved.

2.2. Perfect large eddy simulation. In numerical simulations, the continuous form of (2.1) is replaced by a discretized formulation, which can for instance be written as

$$U_t + R(F(U)) = 0, \tag{2.3}$$

with R denoting a discrete divergence operator applied to the fluxes $F = F^c - F^v$. It is clear that the discretization operator is consistent in the limit $h \to 0$, i.e., if (2.3) is solved on a grid with negligible discretization errors, the continuous solution U is recovered. For turbulent problems, (2.3) often cannot be solved on such a fine grid, due to the multi-scale character of turbulence and the consequently prohibitive computational cost. The methodology of large eddy simulation (LES) avoids to resolve the expensive fine-scale components of the solution by only resolving its coarse scales. This corresponds to applying a low-pass filter $\overline{(\cdot)}$ with linear filter kernel to (2.3). Under the common assumption that filter and time derivative commute, this yields

$$\overline{U}_t + \overline{R(F(U))} = 0 \tag{2.4}$$

as the exact evolution equation for the coarse-scale solution $\overline{U}$. Note that, although (2.4) is a coarse-scale formulation, the closure problem has not been solved yet: Computation of the second term in (2.4) would require the knowledge of U (on a DNS level), since F depends non-linearly on it. For practical LES, $\overline{U}$ is advanced in time using an appropriate numerical scheme with its spatial discretization operator $\tilde{R}(\overline{U})$ applied to the coarse-scale solution. Note that $\tilde{R} \to R$ in the above limit of $h \to 0$, but that $\tilde{R}$ introduces approximation errors otherwise. Further, $\tilde{R}$ can be non-linear. In fact, discretization operators typically contain some form of non-linearity to ensure stability in under-resolved regions. We use this notation to highlight that while the filter operator $(\bar{\cdot})$ is known analytically, the discretization induced filtering $(\tilde{\cdot})$ is not. Equation (2.4) can then be written as

$$\overline{U}_t + \tilde{R}(\overline{U}) = \underbrace{\tilde{R}(\overline{U}) - \overline{R(F(U))}}_{\text{perfect LES closure}}. \tag{2.5}$$

The described approach yields the perfect LES closure term as the right-hand side of (2.5).[1] If the closure term $(\tilde{R}(\overline{U}) - \overline{R(F(U))})$ is known during simulation (e.g., computed from high-fidelity DNS data), the *exact* coarse-scale solution $\overline{U}$ can be recovered, as is verified in [3]. Since this DNS data is not available in practical situations, the right-hand side of (2.5) is typically replaced by a model, which is based on physical or mathematical reasoning, to solve this closure problem. A more detailed discussion on the approach of *perfect LES* can be found in [3], we merely wish to emphasize here that this is a formal framework for the analysis of LES.

By establishing (2.5), two distinct choices have to be made. The first is the choice of the LES operator. Since the LES discretization is a constituent part of the perfect closure terms, approximate closure models with a given set of model parameters generally cannot be expected to be optimal for a wide variety of discretizations. An appropriate closure model and its parameters rather have to be chosen discretization-specific by design. The second and more obvious choice is the selection of the LES filter, which directly defines the coarse-scale space and the unknown component of the closure term $\overline{R(F(U))}$. More importantly, the chosen filter function also predefines the coarse-scale solution $\overline{U}$ itself, and thus the overall target quantity of LES. Altering the LES filter therefore not only changes the closure terms, but also the exact coarse-scale solution.

REMARK 2.1. It is important to stress that, if the operator $\tilde{R}$ is linear, the distinction between filtering the discrete or the continuous equations becomes expendable, since in this special case the *linear* filter and the *linear* operator do, in fact, commute; disregarding possible issues with violation of homogeneity on stretched grids. This is also typically the case, if the coarse-scale solution is perfectly resolved by the numerical scheme and the closure term does not introduce subfilter scales, since then the numerical approximation error becomes negligible. Under these assumptions, also the discretization-dependence of approximate closure models vanishes. Note that this special case is labeled *explicitly filtered* LES. Here, the scale separation filter and the discretization are completely independent from each other, and spatial convergence of the filtered solution is simply achieved by grid refinement. The opposite approach is the *implicitly filtered* LES, in which discretization and filter are intrinsically linked and the discretization operator defines the coarse-scale solution. This has tremendous advantages in terms of computational efficiency, but makes analysis very tough. Our chosen approach of the perfect LES is an attempt to improve this situation. We recognize that some (unknown) filtering is introduced by the numerical scheme that defines the expected solution and the closure terms. Note that we cannot specify a filter kernel, but we can make educated

[1]Note that (2.5) is written with the practical application of an implicitly filtered LES in mind. The explicitly filtered LES formulation can be derived by applying an additional $(\bar{\cdot})$ filter to (2.5). In this case, grid refinement under the filter will ensure $\tilde{R} \to \bar{R}$.

guesses for what the kernel might look like, as is discussed in Section 2.4. We also incorporate the fact that we do know that spatial discretization is applied, i.e., we consider the effects of a non-errorfree operator $\tilde{R}$ in Section 2.4.

2.3. Homogeneous isotropic turbulence. To investigate the nature of the LES closure, it can be exploited that the perfect closure terms can be computed exactly from high-fidelity DNS data, as indicated in Section 2.2. To this end, decaying homogeneous isotropic turbulence (DHIT) is investigated, which is a common test case for LES closure models and is studied extensively in the literature, e.g., [29]. For this test case, a periodic box is initialized with a pseudo-turbulent flow field. The transfer of kinetic energy from lower to higher wavenumbers and the viscous dissipation then lead to a decay of kinetic energy over time, as is shown in Figure 2.1.

The DHIT simulations employ a cubic domain of size $[0, 2\pi]^3$ with periodic boundary conditions. Random pseudo-turbulent flow field realizations for a prescribed spectral distribution of kinetic energy proposed by Chasnov [5] are used as initial conditions. This initial energy spectrum is given by

$$E_{\text{kin}}(k, t = 0) = \frac{1}{2}\, a_s\, u_0^2\, k_p^{-1} \left(\frac{k}{k_p}\right)^s \exp\left[-\tfrac{1}{2} s \left(\frac{k}{k_p}\right)^2\right],$$

for which the parameters are chosen as $s = 4$, $u_0 = 5$, $k_p = 4$, $a_s = 1.0638$ and which is displayed in Figure 2.1. The procedure proposed by Rogallo in [28] allows to derive arbitrary many incompressible velocity fields which follow the given kinetic energy spectrum and exhibit a root mean square velocity of unity. To obtain a compressible flow state from the incompressible velocity field, the mean pressure of the initial flow field is chosen in order to obtain a Mach number of 0.1 with respect to the maximum velocity in the flow field. The pressure fluctuations are then set thermodynamically consistent to the prescribed flow field with unit density. The Reynolds number with respect to the Taylor microscale is approximately $Re_\lambda \approx 180$ at the onset of exponential energy decay. All time specifications in the following are normalized by unit length and unit velocity. This test case of "turbulence in a box" can be seen as the building block of turbulence in the absence of boundary conditions and is thus highly suitable to study turbulent dynamics and model development.

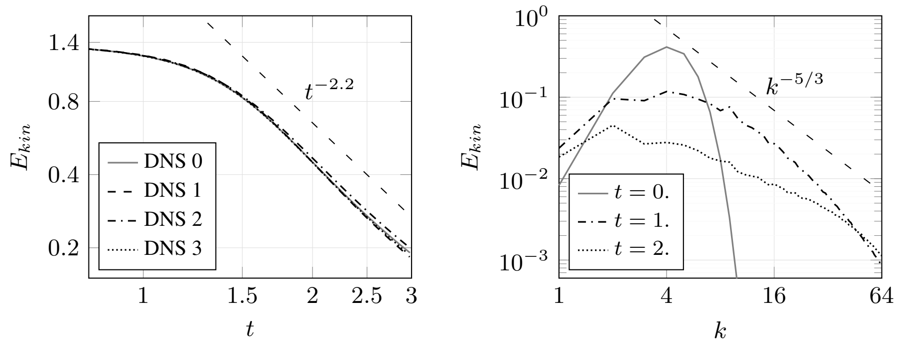

FIG. 2.1. *The temporal evolution of the integral kinetic energy over time for several DNS of the DHIT test case (left) and the distribution of kinetic energy over the wavenumbers k at distinct time instants for a single DNS (right).*

The DNS of the test case were conducted with the discontinuous Galerkin solver FLEXI [17] on the supercomputer Hazel Hen at the High-Performance Computing Center

Stuttgart (HLRS). An eighth-order DG scheme was used on a Cartesian mesh with 64^3 elements, yielding around 134 million degrees of freedom per solution variable. The computational cost for a single DNS up to $t = 3$ was around 15,000 CPU-hours. In addition to the solution U, also the term $R(F(U))$ was stored during the simulation.

2.4. The exact closure terms. The DNS solution is filtered in a post-processing step to obtain the coarse-scale quantities, i.e., the coarse-scale solution $\overline{U}$ and the filtered DNS divergence $\overline{R(F(U))}$. Three distinct filter shapes are investigated in this work. As discussed above, the actual filter that corresponds to a discretization is unknown and typically nonlinear and inhomogeneous, thus, we choose filter forms associated with typical discretization operators. For all filters, the solution is eventually represented in the underlying discontinuous Galerkin (DG) framework. The solution $\overline{U}$ is thus approximated in each element by means of a polynomial basis

$$U \approx \sum_{i,j,k=0}^{N} \hat{U}_{ijk}\, \varphi_{ijk}(x), \tag{2.6}$$

with the coefficients $\hat{U}_{ijk}$ and the three-dimensional polynomial basis functions $\varphi_{ijk}(x)$. In this work, Lagrange polynomials of degree N on Gauss-Lobatto interpolation points are used as basis functions with a tensor-product ansatz, which is a common representation for DG methods. For more details, the reader is referred to [17]. The first filter arising naturally from this DG framework is a local L_2-projection onto piecewise polynomials. Accordingly, the DNS solution is projected element-wise on a coarse LES grid with 8^3 equispaced elements, each comprising a local polynomial basis of degree $N = 5$. This results in $8(N + 1) = 48$ degrees of freedom in every spatial direction. The coefficients of the polynomial representation for the coarse-scale solution can thus be obtained from the DNS solution U by performing the L_2-projection onto the basis functions $\varphi_{ijk}(x)$ for each element E as

$$\hat{\overline{U}}_{ijk} := M^{-1} \int_E U(x)\, \varphi_{ijk}(x)\, \mathrm{d}V, \tag{2.7}$$

with M the mass matrix of the DG scheme. This filter can be seen as an approximation to the true DG filter function in the limit for $N \to \infty$, without regarding the interface coupling between the individual elements.

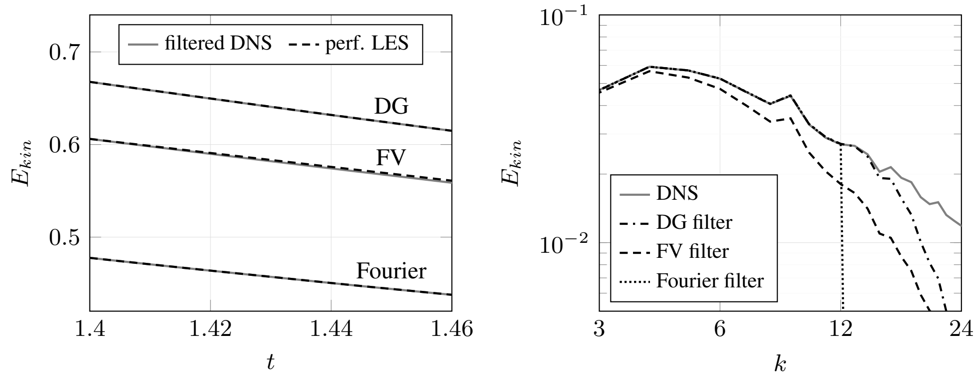

FIG. 2.2. *Left: temporal evolution of kinetic energy for the filtered DNS and the perfect LES for the FV, DG, and Fourier filter, respectively. Right: energy spectra at $t = 1.4$ for the DNS solution and the different LES filters.*

Secondly, the DNS solution is filtered by an L_2-projection onto a piecewise constant representation on a Cartesian mesh with 48^3 elements, again resulting in 48 degrees of freedom in every spatial direction. This filter can be interpreted as a finite volume (FV) flavored representation of the solution (sometimes called box filter) and directly follows from (2.6) and (2.7) for $N = 0$, i.e., for a piece-wise constant basis. In contrast to the DG filter, this representation is thus only first-order accurate. Due to the reduced number of points per wavelength resolution capability [10], the spectral distribution of kinetic energy in the coarse-scale solution exhibits greater deviations from the full DNS solution than the DG representation does, as shown in Figure 2.2.

In contrast to the first two local LES filters, the third examined LES filter is a global Fourier cutoff filter. Hence, a fast Fourier transform (FFT) is applied to the DNS solution yielding a solution representation similar to (2.6), but with respect to a global Fourier basis instead of an element-wise polynomial basis. Then, a cutoff filter at wavenumber $k_{\max} = 12$ is applied as LES filter. This yields a truncated solution $\overline{\mathscr{U}}$ in Fourier space containing the first $k_{\max}$ global Fourier modes of the full solution, while the higher modes $k > k_{\max}$ are set to zero. The coefficients of the filtered solution with respect to the global Fourier basis are thus given as

$$\hat{\overline{\mathscr{U}}}_k = \begin{cases} \hat{\mathscr{U}}_k & \text{if } k \leq k_{\max}, \\ 0 & \text{otherwise.} \end{cases}$$

The DG representation with $N = 5$ discussed above is then used to represent the filtered solution again in physical space. The resolution of this DG discretization ensures that all wavelengths present in the solution are represented accurately by the underlying DG representation, as can be seen in Figure 2.2. This choice of a globally defined filter and a discretization scheme that resolves all occurring scales with only negligible error corresponds to the case of the explicitly filtered LES described above.

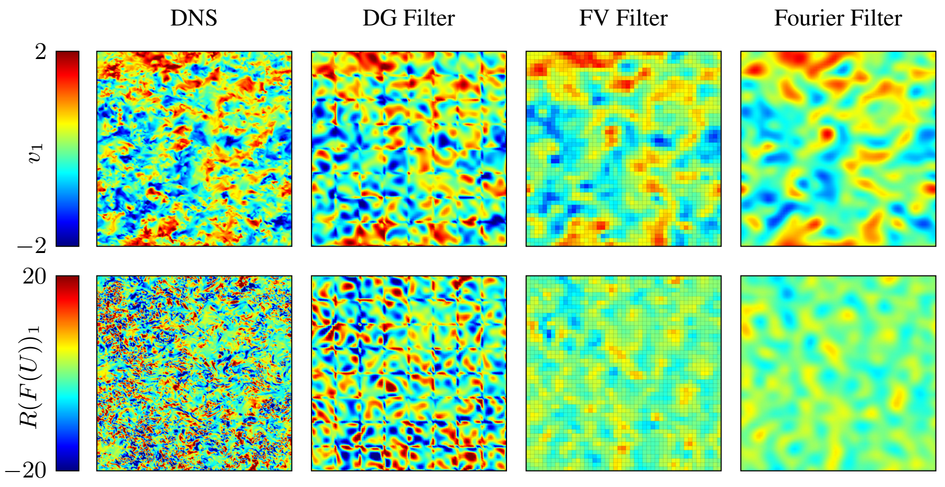

FIG. 2.3. *Two-dimensional slices of the full and the filtered x-velocity field (top) and flux divergence (bottom) at $t = 1.4$. Shown is the field solution (from left to right) for the DNS, the local DG filter, the local FV filter, and the global Fourier filter.*

The three chosen filter types thus cover the range of typically used LES methods: from the global filtering associated with pseudospectral methods to local filter operations onto a

ETNA
Kent State University and
Johann Radon Institute (RICAM)

polynomial subspace with the FV discretization as the limit in terms of locality. The full and the filtered field solution of the x-velocity v_1 and the DNS operator $R(F(U))_1$ are shown in Figure 2.3 for all three filter forms. It is important to stress that only the Fourier filter is strictly homogeneous, for which Moser et al. [24] argue that discretization effects do not have to be modeled, since differentiation and filtering commute. In its own right, Figure 2.3 already reveals some interesting insights. Firstly, as expected, the choice of the filter $(\bar{\cdot})$ defines the resolved field $\bar{U}$ as a coarse-scale representation of the full solution. The resolved fields show different properties in terms of locality and smoothness as caused by the filter. Even more striking, however, is the difference in the closure terms induced by the filter, underlining the statement that the optimal closure is a direct function of the LES filter form; and thus the discretization operator and its properties.

It was verified that the derived closure terms are indeed perfect in the sense that they recover the filtered DNS solution in every timestep. For this, we performed what we refer to as *perfect LES*. This means that (2.5) is solved numerically with a given discretization scheme $\tilde{R}(\cdot)$ and with a filtered DNS state $\overline{U}$ as initial condition. In this work, a third-order Adams-Bashforth method with a fixed timestep of $\Delta t = 10^{-4}$ is used for time integration and the simulation is run for $t \in [1.40, 1.46]$. For each of the simulation timesteps, the exact closure terms, i.e., the right-hand side of (2.5), are computed in a preprocessing step from DNS data for the given LES filter and discretization. In each timestep of the simulation, these exact closure terms are then read from disk and added into the simulation. A more throughout discussion of this methodology can be found in [3]. The results of these simulations in Figure 2.2 demonstrate that the obtained coarse-scale solutions indeed coincide with the respectively filtered DNS solution for all considered filter forms. This confirms our definition of a *perfect* LES and of *perfect* closure terms.

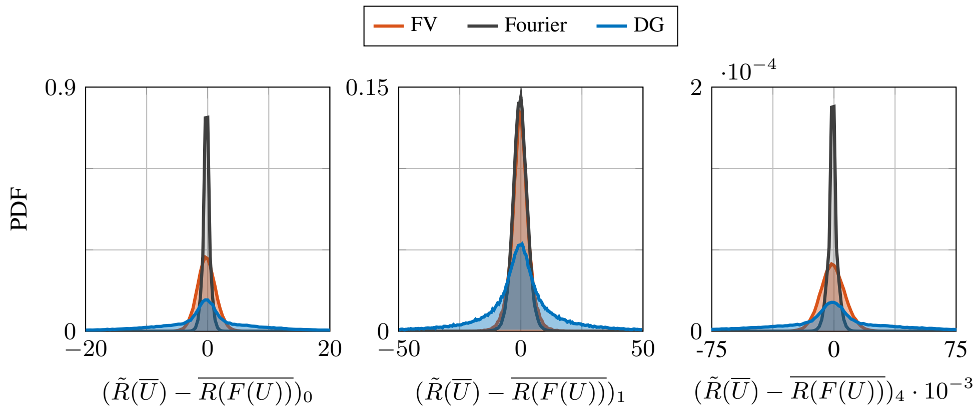

FIG. 2.4. *Histograms of the magnitude of the full closure terms at* $t = 1.4$. *Shown are the distributions of the DG filter, the FV filter, and the Fourier filter for the closure terms of the mass equation (left), the x-momentum equation (center), and the energy equation (right). The histograms are normalized to integrate to unity.*

Further examination of the closure terms reveals that their statistics strongly depend on the chosen LES filter, as presented in Figure 2.4. While the Fourier and FV filter show similar distributions of the closure terms in the momentum equation, the distribution of the DG-filtered data exhibits significantly higher variance. This is caused by the element-wise L_2-projection, which leads to large localized discontinuities in the solution at the element interfaces, c.f. Figure 2.3. These discontinuities lead to large flux contributions by the interface coupling and

ETNA
Kent State University and
Johann Radon Institute (RICAM)

therefore to an overall increase in magnitude for the closure terms. This again stresses the influence of the discretization on the LES closure terms for the implicitly filtered LES approach.

REMARK 2.2. A common assumption for low Mach number flows is that the closure terms of the mass and energy equation become negligible in comparison to the momentum closure, see, e.g., [9]. While this assumption is typically derived in context of the continuous formulation, the assumption might be invalid if discretization effects must be taken into account. As shown in Figure 2.4, while the closure terms for the Fourier-filtered approach for the mass and energy equations are generally very close to zero as denoted by the sharp peak, this is neither the case for the DG nor for the FV filter, which feature rather wide tails. Interestingly, the momentum closure terms for the FV and Fourier filter are similar in shape, while the DG data again shows a much wider distribution. We attribute this behavior to the hybrid nature of the DG operator and the high order approximation used here. Our analysis thus indicates that the LES filter (and the discretization scheme) does influence the validity of the discussed assumption. This notion is also supported by Moser et al. [24]. In order to test these propositions, the perfect LES computations from Figure 2.3 were repeated, but instead of closing all five equations, only the momentum equations are closed in the perfect LES, while leaving the mass and energy equation unclosed. For the Fourier filter, the results showed only negligible deviations from the previous computation, and still matched the filtered DNS data, as expected for the explicitly filtered approach. In contrast, the solution for the DG filter diverged quickly and irrefutably from the filtered DNS data. This demonstrates that the assumption that the closure terms in mass and energy equation are negligible for low Mach number flows does hold for explicitly filtered LES, but not necessarily for implicitly filtered LES, where discretization effects become important. In the following, only the closure of the momentum equations is considered. Note however, that the results presented in Section 5 also hold if instead the full closure, i.e., including the mass and energy equation, is considered as target quantity. This was verified by training the GRU3 network on the DG data for the full closure vector, which achieved similar accuracy to the reported case.

3. Artificial neural networks.

Artificial neural networks (ANN) are general function approximators, which can approximate any continuous functional relationship $Y = f(X)$ between some input data X and output data Y. This notion is supported by the variety of universal approximation theorems for ANN in literature, e.g., [7, 15, 21]. The wide diversity of applications for ANN gave rise to a certain richness in network architectures proposed in literature, with each network type optimized for a specific range of tasks, and a specific nature of data. For our application, we confine ourselves to the discussion of the most basic ANN architecture, the multilayer perceptron in Section 3.1, and the gated recurrent unit (GRU) as a representative of recurrent neural networks (RNN) for series modeling in Section 3.2.

3.1. Multilayer perceptron.

A multilayer perceptron (MLP) consists of multiple *layers*, with each layer comprising a number of *neurons*, as depicted in Figure 3.1. The layers that are not directly connected to the input or output of the network are called *hidden layers*. Each neuron j in layer n takes the outputs of the previous layer's neurons as inputs x_i^n, computes the weighted sum of the inputs with the weights $\mathcal{W}_{ij}^n$, and adds a bias b_j^n. Applying a non-linear *activation function* f_a then yields the neuron's output y_j^n as

$$y_j^n = f_a\left(\sum_i \mathcal{W}_{ij}^n\, x_i^n + b_j^n\right) \qquad \text{with} \quad x_i^n = y_i^{n-1}.$$

Common choices for the activation function are the sigmoid function ($\sigma_s = 1/\left(1+e^{-x}\right)$), the hyperbolic tangent ($\sigma_t = \tanh\left(x\right)$), or the rectified linear unit (ReLU, $\sigma_r = \max\left\{x, 0\right\}$).

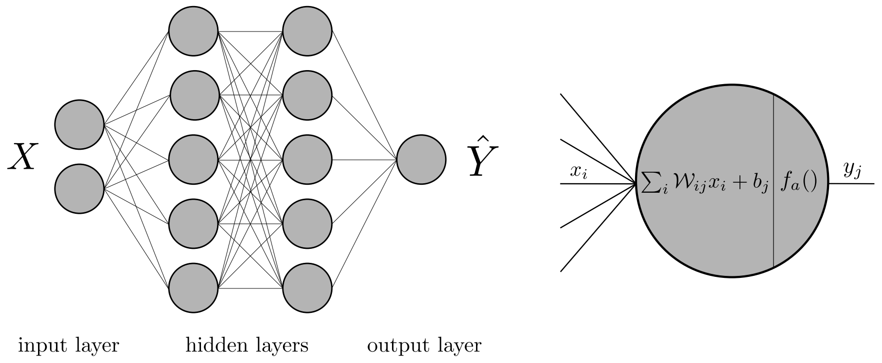

FIG. 3.1. *Shown on the left is the schematic architecture of an MLP with two hidden layers comprising 5 neurons each. Shown on the right is the layout of a single neuron j with the layer index n dropped for clarity.*

The network parameters (i.e., the weight matrices and biases) are generally initialized in a stochastic manner, e.g., with the initialization method proposed by Glorot and Bengio [11] or the method by He et al. [12], which was proposed for networks using the ReLU activation function. Given a dataset of training samples, where each sample consists of an input vector X and the corresponding ground truth Y, the network parameters are optimized to minimize a predefined *loss function*. The loss function is a measure of distance between the network's output $\hat{Y}$ and the ground truth Y for a given training sample and thus quantifies the accuracy of the network's predictions. In the context of ANN, this optimization process is commonly referred to as *training* or *learning* and is achieved by computing the gradient vector of the loss function with respect to the weights efficiently by means of *backpropagation*. Training a neural network therefore refers to finding a set of network parameters that minimizes the loss function for a given training set.

3.2. Gated recurrent units. Recurrent neural networks (RNN) are the state-of-the-art architecture for sequential data, i.e., when the ordering of the input data is important. The underlying principle of RNN is illustrated in Figure 3.2. In addition to the current input sample x_n, RNN receive their last inner state h_{n-1} as input, which allows them to retain information from previous input samples. Different types of RNN are known in the literature, which differ predominantly in the way the new inner state h_n and the output y_n are computed. The standard (*vanilla*) RNN works like an MLP when unrolled in time. The weight matrix and bias are applied to the input (i.e., the current sample and the layer's previous output), followed by an activation function to obtain the state for the next sample. Important to note is that in contrast to an MLP, the weights of an RNN are shared for all time steps. For long input sequences, the RNN concept leads to very deep networks in time, which can cause the gradients to vanish or explode during backpropagation, as is discussed in [4]. More sophisticated RNN architectures alleviate this problem by introducing a gating mechanism that determines the flow of information through the cell in the forward-pass and thus also the propagation of errors during backpropagation. Common representatives of this approach are the long short-term memory (LSTM) architecture proposed by Hochreiter and Schmidhuber in [14] and the gated recurrent unit (GRU) proposed by Cho et al. [6].

The layout of a GRU cell is shown in Figure 3.2. The GRU receives two inputs, the current sample x_n and the previous state of the cell h_{n-1}. The new state h_n is an affine combination of the old state h_{n-1} and a state candidate $\tilde{h}_n$ with

$$h_n = (1 - z_n) \odot h_{n-1} + z_n \odot \tilde{h}_n.$$

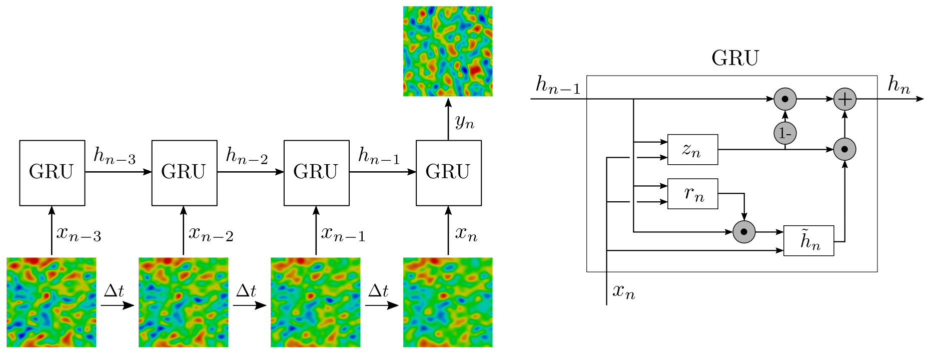

FIG. 3.2. *Shown on the left is the schematic of a GRU unrolled in time. At each time instant, the sample x_i and the hidden state from the previous step h_{i-1} are fed into the GRU, yielding the new state $h_i = f(h_{i-1}, x_i)$. For the GRU architecture, the hidden state coincides with the layer output, i.e., $y_n = h_n$. The detailed architecture of a GRU cell is shown on the right. Here, all indicated operations are performed element-wise.*

Here, $\odot$ indicates element-wise multiplication. Important to note is that for the GRU, the inner state coincides with the layer output, i.e., $h_n = y_n$. This is the major difference between the GRU and the LSTM architecture, since the latter discriminates between the hidden state and the output. The coefficient z_n is determined by the *update gate*, which computes

$$z_n = \sigma_s\left(\mathcal{W}_z x_n + \mathcal{U}_z h_{n-1} + b_z\right), \tag{3.1}$$

with $\{\mathcal{W}_z, \mathcal{U}_z, b_z\}$ as trainable parameters, which are determined during the optimization process. The sigmoid function σ_s ensures $z_n \in [0, 1]$. The state candidate $\tilde{h}_n$ is given by

$$\tilde{h}_n = \sigma_t\left(\mathcal{W}_h x_n + \mathcal{U}_h\left(r_n \odot h_{n-1}\right) + b_z\right),$$

with σ_t as the hyperbolic tangent and $\{\mathcal{W}_h, \mathcal{U}_h, b_h\}$ as trainable parameters. The *reset gate* computes r_n, which determines how much of the previous state is incorporated into the new state candidate. Analogously to (3.1), for the reset gate follows

$$r_n = \sigma_s\left(\mathcal{W}_r x_n + \mathcal{U}_r h_{n-1} + b_r\right).$$

Throughout this work, we will restrict ourselves to GRU networks, since they showed to be easier to train and consistently outperformed the LSTM architecture in terms of accuracy for our application.

4. Training the networks.

In the following experiments, we use ANN with MLP and GRU architecture to learn the underlying non-linear mapping between the coarse-scale velocity vector $[\overline{v}_1, \overline{v}_2, \overline{v}_3]^T$ as input and the filtered DNS operator $\overline{R(F(U))}_{1,2,3}$ as the target quantity, since this is the only unknown contribution to the closure terms. The MLP networks additionally receive the LES operator $\tilde{R}(\overline{U})_{1,2,3}$ of the momentum equations as input, which improved the prediction accuracy considerably, as was also reported for convolutional networks in [3]. The input vector for the MLP is therefore six-dimensional, containing the three velocity components as well as the three components of the LES operator. For the GRU network, however, including the LES operator to the input vector only results in minor improvements, which do not justify the substantial increase in memory consumption. The GRU and MLP networks receive and predict only pointwise data. For all considered networks, the input features are scaled to zero mean and unit variance on the training set. To obtain training data, 10 DNS runs of the DHIT test case with different initial conditions, drawn from the same distribution, are computed. Thereof 9 runs are used for training and validation, while a single run is kept hidden as test set to evaluate the models' performance on unseen data. The training data is obtained by sampling the simulation results between $t = 1.0$ and $t = 1.9$ with an interval of $\Delta t = 0.1$. The isotropy of the flow can be exploited for data augmentation by cyclic interchange of the velocity components v_1, v_2, v_3 in the different spatial directions. This allows to triple the amount of training data to about 30 million point-wise training samples for each LES filter. All networks are trained and evaluated separately for each particular LES filter.

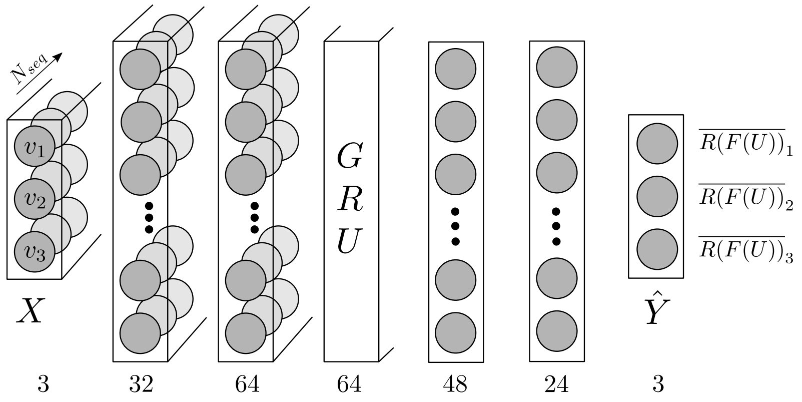

FIG. 4.1. *Architecture of the GRU network. The network consists of 5 hidden layers, with the number of neurons denoted below each layer. The first two hidden layers are executed independently for each sample in the sequence, but with the same weights. Using the many-to-one prediction mode, the GRU layer computes a single output vector from the received inputs, as also shown in Figure 3.2.*

The networks investigated in this work follow the sequential architecture of Figure 4.1 with 5 hidden layers. The first two layers are dense layers with 32 and 64 neurons, followed by a GRU layer with 64 cells and again two dense layers with 48 and 24 neurons, respectively. For the MLP network, the GRU layer is replaced by a dense layer comprising 64 neurons. All layers except the GRU layer employ ReLU activation functions and are initialized with the method proposed by He et al. [13]. For the GRU network, also the influence of different sequence lengths and sampling frequencies for the input data is investigated, with an overview of the different parameter combinations given in Table 4.1. The time increment between input samples is deliberately chosen in the order of magnitude of a stable numerical timestep for the LES computation, which is around $\Delta t \approx 10^{-4}$, in order to examine the feasibility of the described approach for a priori predictions in practical LES. Thus, the predictions of the GRU networks are based on short-time dynamics in the order of the smallest resolved feature.

TABLE 4.1
Characteristics of the different input sequences for the GRU network. The total number of samples per sequence is given by N_{seq}, the time increment between two samples is Δt_{seq}, and the total time interval of the sequence, i.e., the time interval between the first and the last sample, is denoted as Δt_{tot}.

	GRU1	GRU2	GRU3
N_{seq}	3	10	21
Δt_{seq}	$1 \cdot 10^{-3}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$
Δt_{tot}	$2 \cdot 10^{-3}$	$9 \cdot 10^{-4}$	$2 \cdot 10^{-3}$

The mean squared error is used as loss function for all networks and Pearson's correlation coefficient is used as additional performance metric. The correlation coefficient between two quantities a, b is defined as

$$\mathcal{CC}(a,b) = \frac{\mathrm{Cov(a,b)}}{\sqrt{\mathrm{Var(a)}}\sqrt{\mathrm{Var(b)}}}, \tag{4.1}$$

with the covariance $\mathrm{Cov}(\cdot)$ and the variance $\mathrm{Var}(\cdot)$. In addition, the mean L_2-error of the prediction is reported, which is obtained by integrating the prediction errors element-wise on the coarse mesh and averaging the error over all elements.

The whole framework is implemented in TensorFlow 2.1 [1]. The training was carried out on an Nvidia K40c GPU. For all networks, the Adam algorithm by Kingma and Ba [16] is used for optimization. The networks are trained for 50 epochs with a batch size of 256 and an initial learning rate of 0.001, which is halved every 10 epochs. No additional regularization is employed during training, since the training set was sufficiently large and no overfitting is observed by comparing the losses on training and validation set.

5. Results.

The results are split into three parts. In Section 5.1, the training performance and the prediction accuracy of the networks on the test set for each LES filter are discussed. The generalization abilities of the investigated networks across different LES resolutions and filters are investigated in Section 5.2. Finally, the accuracy of the full ANN-based closure terms is examined and compared to common analytical closure models in Section 5.3.

5.1. Training results.

The GRU as well as the MLP architectures are able to learn the unknown closure terms from the coarse-scale input data for all three considered filter forms, whereby the GRU networks outperformed the MLP architecture. A detailed overview of the networks' performance for the different data sets is given in Table 5.1 and a comparison of the predicted field solutions is shown in Figure 5.1. The predicted closure terms of the GRU3 network show excellent agreement with the exact closure terms for all considered filter forms. In addition, the GRU3 network recovers the statistical distribution of the exact closure terms almost perfectly. The MLP architecture, however, cannot reproduce the exact distribution, but exhibits much lower variance in its predictions. The MLP therefore consistently underestimates the occurring extrema of the closure terms. This effect is most pronounced for the DG-filtered data, leading to only 24% correlation between the MLP prediction and the exact terms.

The sampling of the input sequence showed to be of only minor importance for the accuracy of the GRU networks. The GRU3 network shows the best performance for all considered datasets, since it receives all 21 available timesteps and therefore the most extensive temporal information. However, the performance of the GRU1 and GRU2 networks show to be only slightly inferior for most considered cases. Only for the DG-filtered data, the GRU2 shows significantly higher prediction errors than the other two networks. A possible explanation for the similar prediction performance is that for the low Mach number test case, the numerical timestep (which is chosen as time increment in the input sequences) is small in comparison to the relevant physical time scales. Due to the extensive temporal resolution, the larger timestep of the GRU1 is still sufficient to resolve the relevant physical phenomena and therefore to accurately predict the LES closure terms. This finding can be exploited to reduce the temporal resolution in the input sequence during training, since storing large amounts of DNS data is prohibitive in terms of storage capacity. Further experiments indicate that the prediction accuracy of the GRU networks could be improved further by increasing the network depth with additional GRU layers. However, the improvements with each additional layer diminish quickly while the computational effort increases. Moreover, the networks are deliberately chosen small with only around 32,000 parameters to allow their efficient

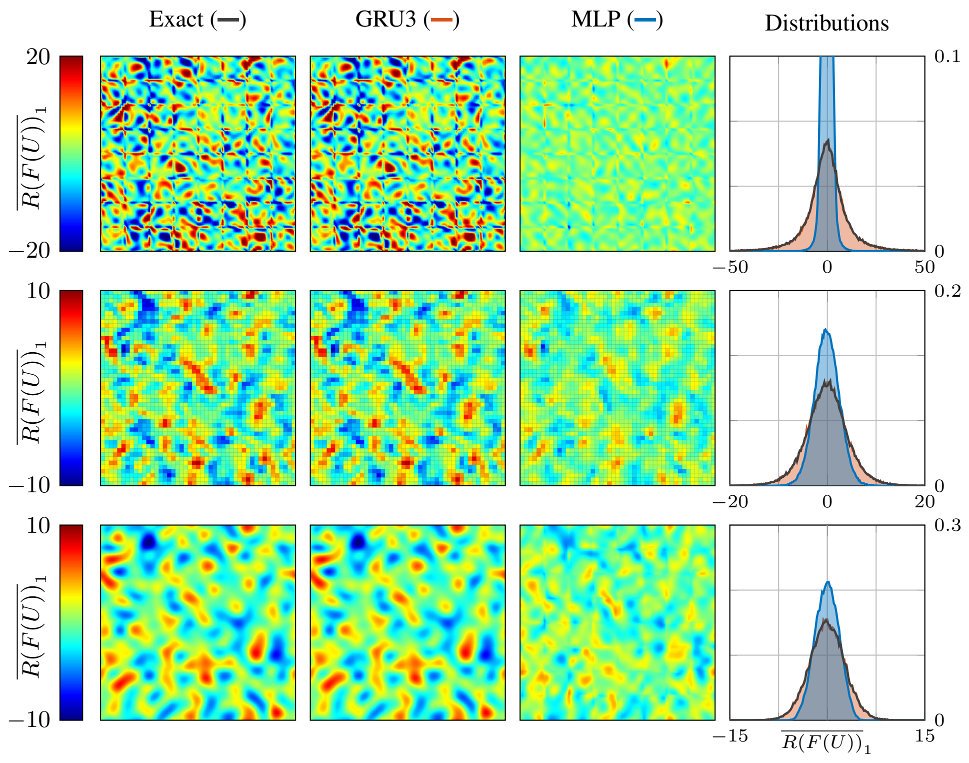

FIG. 5.1. *Two-dimensional slices of the exact closure term of the x-momentum equation with the predictions of the GRU3 and the MLP architecture for the DG filter (top), the FV filter (center), and the Fourier filter (bottom). The distributions of the exact and the predicted closure term are shown on the very right. The distributions are normalized to integrate to unity.*

evaluation for potential applications in data-driven closure models for practical LES. To address the importance of closing the mass and energy equation for the DG data, which is discussed in Remark 2.2, it was investigated whether the GRU networks can also predict the closure terms for the full equation system. Accordingly, the input vector for the GRU networks was extended by the density ρ and the energy e. After training the network, it was able to predict the whole closure vector with all 5 entries up to a similar degree of accuracy as reported here for the momentum equation.

The results presented in this section are rather remarkable. They indicate that for this canonical turbulent flow, the *exact* closure terms can be predicted by a recurrent neural network with (likely almost) *arbitrary accuracy* for a range of different filter forms. The achieved error norms and correlations vastly outperform previously reported data, see, e.g., [3] for results with convolutional neural networks. It should also be stressed that the exact evaluation of the closure terms requires knowledge of the full fine scale solution U, while the ANN-based approaches predict from the coarse-scale quantities $\bar{U}$ only. This astonishing capability of the method used, which in a sense seems to entail some form of implicit filter inversion, will be investigated further in the future.

5.2. Generalization. To test the generalization abilities of the networks, the trained GRU3 networks from Section 5.1, which are each trained for one particular LES filter, are applied to the test data of the other filter functions, which they have not seen during training. The results given in Table 5.2 indicate, that all networks generalize well across the different

TABLE 5.1
Prediction accuracy of the trained ANN on the respective test sets. The GRU1 and GRU2 networks for the Fourier data are trained with halved initial learning rate, as they show divergent behavior otherwise. The best performance on each dataset is highlighted in bold font.

	DG Filter		FV Filter		Fourier Filter	
	L_2-Error	$\mathcal{CC}$	L_2-Error	$\mathcal{CC}$	L_2-Error	$\mathcal{CC}$
MLP	$2.56 \cdot 10^{+2}$	0.2456	$1.02 \cdot 10^{+2}$	0.5744	$3.08 \cdot 10^{+1}$	0.6712
GRU1	$9.44 \cdot 10^{-1}$	$\mathbf{0.9989}$	$2.53 \cdot 10^{-1}$	$\mathbf{0.9992}$	$1.09 \cdot 10^{-1}$	0.9992
GRU2	$1.07 \cdot 10^{+2}$	0.8157	$2.33 \cdot 10^{-1}$	$\mathbf{0.9992}$	$9.25 \cdot 10^{-2}$	$\mathbf{0.9993}$
GRU3	$\mathbf{9.14 \cdot 10^{-1}}$	$\mathbf{0.9989}$	$\mathbf{2.31 \cdot 10^{-1}}$	$\mathbf{0.9992}$	$\mathbf{9.15 \cdot 10^{-2}}$	$\mathbf{0.9993}$

TABLE 5.2
Prediction accuracy of the GRU3 networks for each LES filter. The rows refer to the datasets the networks were originally trained on and the columns refer to the test data the networks are evaluated on.

	DG Data		FV Data		Fourier Data	
	L_2-Error	$\mathcal{CC}$	L_2-Error	$\mathcal{CC}$	L_2-Error	$\mathcal{CC}$
GRU3-DG	$9.14 \cdot 10^{-1}$	0.9989	$3.60 \cdot 10^{-1}$	0.9988	$2.57 \cdot 10^{-1}$	0.9979
GRU3-FV	$1.46 \cdot 10^{0}$	0.9867	$2.31 \cdot 10^{-1}$	0.9992	$9.49 \cdot 10^{-2}$	0.9993
GRU3-Fourier	$9.15 \cdot 10^{0}$	0.8888	$3.79 \cdot 10^{-1}$	0.9988	$9.15 \cdot 10^{-2}$	0.9993

filter forms. Especially for the FV and Fourier data, all networks still achieve over 99% correlation between the predicted and the exact closure terms. Only for the DG-filtered data, the prediction accuracy of the network that is trained on the Fourier data drops significantly. This is expected, since the distribution of the closure terms for the DG filter exhibits much larger variance than the Fourier data. Since the network has never learned to reproduce closure terms of this magnitude, it systematically under-predicts the extrema of the DG closure terms leading to a less accurate prediction. These results indicate that, despite the statistical differences between the closure terms for different LES filters, the relationship between the temporal evolution of the coarse-scale solution and the corresponding closure terms seems to exhibit a universal character and to transfer reasonably well across the different filter forms. This is not surprising, since the target quantity $\overline{R(F(U))}_i$ defines the temporal evolution of the coarse-scale solution (see (2.4)) and thus, the inverse mapping allows to deduce the filtered flux term $\overline{R(F(U))}_i$ from the temporal evolution of the coarse-scale inputs, which is to some extent independent of the used filter function.

In a following step, the influence of the LES filter width on the GRU3 networks' prediction accuracy is examined. For the implicit LES filters, this corresponds to changing the resolution of the discretization. Thus, the number of elements in each spatial direction is reduced to 75% and 50% of the baseline resolution, on which the networks are trained. This yields a total of 36^3 and 24^3 degrees of freedom for the FV and DG filter, and a cutoff wavenumber of $k_{\max} = 9$ and $k_{\max} = 6$ for the Fourier filter, respectively. Due to the three-dimensional domain, the lowest resolution corresponds to only one eighth of the total degrees of freedom in comparison to the baseline resolution. Each network is only evaluated on the LES filter it is trained on, but the original version trained on the finest grid is used, i.e., the networks are not retrained on the coarser grids. The results in Table 5.3 indicate that all networks are able to generalize across the different resolutions and retain low prediction errors for all examined cases. The exact and the predicted field solution as well as the filtered velocity field

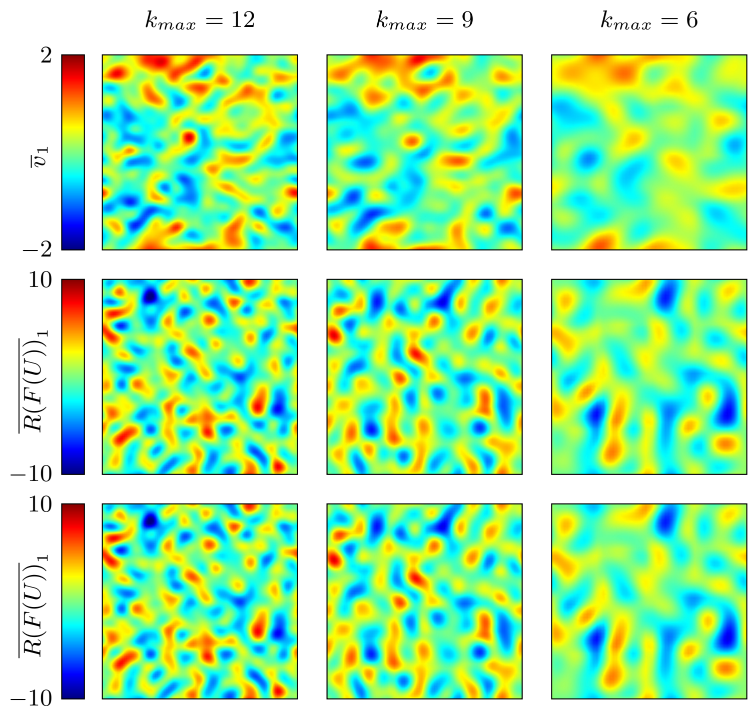

FIG. 5.2. *Filtered x-velocity as network input (top), the exact DNS part of the closure term as network target quantity (center), and the predictions of a single GRU3 network for different cutoff wavenumbers $k_{\max}$ of the Fourier filter (bottom). The network was only trained on Fourier-filtered data with $k_{\max} = 12$.*

TABLE 5.3
Prediction performance of the GRU3 networks on LES data with different resolutions, but with the LES filter the respective networks are trained on. The resolution is quantified by the degrees of freedom (DOF) in each spatial dimension. For the Fourier filter, the baseline resolution of 48 DOF refers to $k_{\max} = 12$ and the lower resolutions to $k_{\max} = 9$ and $k_{\max} = 6$, respectively.

	DG Filter		FV Filter		Fourier Filter	
	L_2-Error	$\mathcal{CC}$	L_2-Error	$\mathcal{CC}$	L_2-Error	$\mathcal{CC}$
48 DOF	$9.14 \cdot 10^{-1}$	0.9989	$2.31 \cdot 10^{-1}$	0.9992	$9.15 \cdot 10^{-2}$	0.9993
36 DOF	$1.15 \cdot 10^{0}$	0.9984	$2.94 \cdot 10^{-1}$	0.9986	$1.23 \cdot 10^{-1}$	0.9989
24 DOF	$1.10 \cdot 10^{0}$	0.9975	$2.19 \cdot 10^{-1}$	0.9984	$1.16 \cdot 10^{-2}$	0.9986

are exemplarily shown for the Fourier-filtered data in Figure 5.2 for the different resolutions. This ability to generalize across different resolutions can be explained by the same arguments as stated above.

5.3. Full closure terms. In a last step, the predictions of the ANN for $\overline{R(F(U))}$ are used to compute the full closure term $\tilde{R}(\overline{U}) - \overline{R(F(U))}$. It seems important to stress again that $\overline{R(F(U))}$ is the only unknown term in the full closure, since the coarse-scale solution $\overline{U}$ and the discretization operator $\tilde{R}(\cdot)$ are known. Moreover, the obtained ANN-based closures are compared to the predictions of the common gradient model (GM) [2] and the static Smagorinsky model (SSM) [33] for the given filtered flow fields. These models are detailed in Appendix A and B, respectively. The results presented in Figure 5.3 and Table 5.4 show that

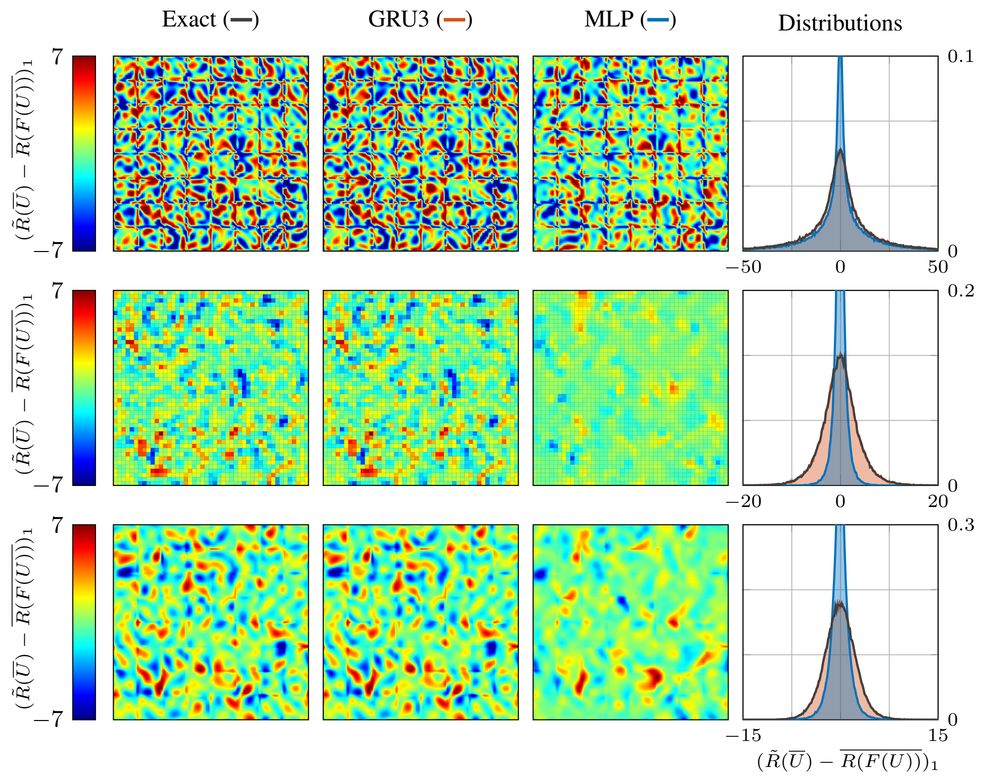

FIG. 5.3. *Slices of the full closure term $\tilde{R}(\overline{U}) - \overline{R(F(U))}$ for the x-momentum equation with $\overline{R(F(U))}$ either (from left to right) computed exactly from DNS, as predictions of the GRU3 models or of the MLP models. The normalized distributions of the closure terms for all three cases are shown on the right. The columns correspond to the DG filter (top), the FV filter (middle) and the Fourier filter (bottom).*

the predictions of the GRU models recover also the full closure term up to very high accuracy. Interestingly, the correlations for the DG data are consistently higher than the ones reported for the DNS term by itself in Section 5.1. For the Fourier and FV filters in contrast, the correlation coefficients are similar. However, the full closures based on the MLP predictions show less correlation than for the DNS term by itself. These differences in behavior stem from the non-linearity of the correlation coefficient in (4.1), which does not guarantee that the sum of two quantities achieves a correlation similar to the correlations of the individual quantities.

The SSM is a functional model, which mainly mimics the energy drain from the resolved to the non-resolved length scales. This model is known to be only poorly correlated with the perfect closure [27]. This also shows in Table 5.4, where the SSM achieves only $|\mathcal{CC}| < 0.2$. While the Smagorinsky model performs poorly in this *a priori* test, it can provide accurate and stable results in practical simulations. In contrast, ANN-based closures are often found to cause instabilities, despite achieving high accuracy in *a priori* tests [3]. This stresses the importance of *a posteriori* analysis of the derived ANN models in future research. The gradient model can be derived as an approximation to the exact subgrid stresses for a Gaussian LES filter [27]. This explains why the model achieves decent accuracy on the data with the smooth Fourier filter, but fails for the DG filter, where the solution and its gradients are discontinuous, as is shown in Table 5.4. Such mismatches appear oftentimes when closure models are derived in an explicitly filtered LES setting but are then applied for implicit LES.

TABLE 5.4
Correlation coefficient $\mathcal{CC}(M_{exact}, M_{ANN})$, where M_{exact} is the exact full closure term $\tilde{R}(\overline{U}) - \overline{R(F(U))}$ computed from DNS data, while M_{ANN} is the full closure term using an ANN-based prediction for $\overline{R(F(U))}$. The results are also compared to the gradient model (GM) and the static Smagorinsky model (SSM), which are detailed in Appendix A and Appendix B, respectively.

	MLP	GRU1	GRU2	GRU3	GM	SSM
DG	0.9267	0.9998	0.9748	0.9998	−0.2318	−0.0926
FV	0.3212	0.9989	0.9990	0.9990	0.1706	−0.1963
Fourier	0.5694	0.9990	0.9992	0.9992	0.4341	0.1709

With such approaches, the discretization effects are not accounted for. These restrictions are alleviated in the proposed machine learning framework, which does not only incorporate the discretization effects by construction, but also allows to transfer the trained ANN to other discretizations and resolutions, as was demonstrated in Section 5.2.

6. Conclusions.

In this work, we have derived a general framework for LES closure models, with the emphasis put on the effects and influences of the discretization operator. In practical LES, the discretization operator itself acts as the scale-separating filter. Its properties (non-linearity, inhomogeneity) can make a rigorous analysis tough, and often lead to the fact that the associated errors (often labeled commutation errors) are ignored or overlooked. As a result of this, closure model development is in some cases "blind" towards the underlying discretization and its induced filter form. This can be problematic, as the optimal closure terms that should govern model development are indeed a function of the filter.

We have investigated the dependence of the closure on the filter functions through the case of decaying homogeneous isotropic turbulence (DHIT), which serves as the simplest canonical test case for turbulence. We have build a DNS database from an ensemble of runs and have computed the exact closure terms for several LES filter functions. For this, we used filter functions that mimic the action of a discretization filter. They are derived from the solution representation of DG and FV schemes. A third, global Fourier cutoff filter represents the explicitly filtered LES approach. We have demonstrated that the computed closure terms are consistent in the sense that they are able to recover the filtered DNS solution in each time step, thus forming the perfect LES model. Different neural networks with MLP and GRU architecture have been trained to predict the unknown contributions of the closure terms for each examined filter function solely from known coarse-scale data. We have demonstrated that the GRU networks are able to approximate the exact closure terms up to a very high degree of accuracy for all considered filter forms. The GRU networks have also been shown to generalize well across different filter functions and filter widths that they have not seen during training.

These highly accurate predictions and the networks' generalization abilities are encouraging results and mark a starting point for the development of universal data-driven LES closure models. Being able to predict the exact closure term that perfectly "fits" both the physical closure problem as well as the discretization effects offers the chance to guide discretization-adapted model development. Future work will be focused on applying the network predictions as closure model during practical LES. Since direct incorporation of the network predictions might cause instabilities, as was found in [3], also strategies to incorporate the ANN predictions into a stable data-driven model have to be examined. Further, the application and extension of the described methodology to other canonical turbulent flow problems, and especially wall-bounded flows, will be investigated.

Acknowledgments. This research was funded by Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC 2075

– 390740016. The authors gratefully acknowledge the support by the Stuttgart Center for Simulation Science (SimTech) as well as the support and the computing time provided by the HLRS through the project "hpcdg".

Appendix A. Gradient model. The gradient model is a common closure model for LES proposed in [2]. It can be derived by performing a Taylor expansion of the subgrid scale stress tensor for a Gaussian LES filter [27]. The model thus reads

$$\tau_{ij}^{GM} := \frac{\overline{\Delta}^2}{12}\left(\frac{\partial \overline{v}_i}{\partial x_k}\frac{\partial \overline{v}_j}{\partial x_k} - \frac{1}{3}\frac{\partial \overline{v}_l}{\partial x_k}\frac{\partial \overline{v}_l}{\partial x_k}\delta_{ij}\right),$$

with the grid spacing $\overline{\Delta}$, which was computed as the domain length divided by the number of degrees of freedom in the respective direction. In this paper, the exact gradients are approximated for all filters by the gradients of the underlying DG representation. This is expected to give reasonable results for the Fourier-filtered case, where discretization effects are negligible and the DG gradients are a good approximation. Generally, these subgrid stresses are then added to the physical stresses in (2.2). For comparison with the ANN-based models, we computed the subgrid forces which result from the subgrid stress predicted by the gradient model.

Appendix B. Smagorinsky model. The static Smagorinsky model was originally proposed by Smagorinsky in [33] and is a common representative of the functional modeling approach. The model mimics the transport of kinetic energy from resolved to non-resolved length scales by introducing an additional eddy-viscosity as

$$\nu_{SGS} := \left(C_s\overline{\Delta}\right)^2\sqrt{2\overline{S}_{ij}\overline{S}_{ij}},$$

with the model parameter $C_s \approx 0.17$ and the filter width $\overline{\Delta}$, which was computed analogously to Appendix A. The filtered rate-of-strain tensor $\overline{S}_{ij}$is defined as

$$\overline{S}_{ij} := \frac{1}{2}\left(\frac{\partial \overline{v}_i}{\partial x_j} + \frac{\partial \overline{v}_j}{\partial x_i}\right).$$

Similar to the gradient model in Appendix A, the gradients of the underlying DG scheme are used to approximate the continuous derivatives. The viscosity can then be obtained from the kinematic viscosity with $\mu_{SGS} = \rho\,\nu_{SGS}$.

REFERENCES

[1] 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. Mane, 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. Viegas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, *TensorFlow: large-scale machine learning on heterogeneous distributed systems*, e-print arXiv:1603.04467, March 2016.
https://arxiv.org/abs/1603.04467.
Software available from https://tensorflow.org.

[2] J. Bardina, J. Ferziger, and W. Reynolds, *Improved subgrid-scale models for large-eddy simulation*, in 13th Fluid and Plasmadynamics Conference 1980, AIAA paper 80-1357, AIAA, Reston, 1980, 10 pages.

[3] A. Beck, D. Flad, and C.-D. Munz, *Deep neural networks for data-driven LES closure models*, J. Comput. Phys., 398 (2019), Art. 108910, 23 pages.

[4] Y. Bengio, P. Simard, and P. Frasconi, *Learning long-term dependencies with gradient descent is difficult*, IEEE Trans. Neural Netw., 5 (1994), pp. 157–166.

[5] J. R. Chasnov, *The decay of axisymmetric homogeneous turbulence*, Phys. Fluids, 7 (1995), pp. 600–605.
[6] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, *On the properties of neural machine translation: encoder-decoder approaches*, in Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, October 2014, ACL, Stroudsburg, 2014, pp. 103–111.
[7] G. Cybenko, *Approximation by superpositions of a sigmoidal function*, Math. Control Signals Systems, 2 (1989), pp. 303–314.
[8] M. Gamahara and Y. Hattori, *Searching for turbulence models by artificial neural network*, Phys. Rev. Fluids, 2 (2017), Art. 054604, 20 pages.
[9] E. Garnier, N. Adams, and P. Sagaut, *Large Eddy Simulation for Compressible Flows*, Springer, Dordrecht, 2009.
[10] G. J. Gassner and A. D. Beck, *On the accuracy of high-order discretizations for underresolved turbulence simulations*, Theor. Comput. Fluid Dyn., 27 (2013), pp. 221–237.
[11] X. Glorot and Y. Bengio, *Understanding the difficulty of training deep feedforward neural networks*, in Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, May 2010, Proceedings of Machine Learning Research, PMLR, 2010, pp. 249–256.
[12] K. He, X. Zhang, S. Ren, and J. Sun, *Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification*, in Proceedings of the IEEE International Conference on Computer Vision (ICCV), December 2015, IEEE Conference Proceedings, Los Alamitos, 2015, pp. 1026–1034.
[13] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016, IEEE Conference Proceedings, Los Alamitos, 2016, pp. 770–778.
[14] S. Hochreiter and J. Schmidhuber, *Long short-term memory*, Neural Comput., 9 (1997), pp. 1735–1780.
[15] K. Hornik, *Approximation capabilities of multilayer feedforward networks*, Neural Netw., 4 (1991), pp. 251–257.
[16] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, e-print arXiv:1412.6980, December 2014. https://arxiv.org/abs/1412.6980
[17] N. Krais, A. Beck, T. Bolemann, H. Frank, D. Flad, G. Gassner, F. Hindenlang, M. Hoffmann, T. Kuhn, M. Sonntag, and C.-D. Munz, *FLEXI: A high order discontinuous Galerkin framework for hyperbolic–parabolic conservation laws*, Comput. Math. Appl., 81 (2021), pp. 186–219.
[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, *ImageNet classification with deep convolutional neural networks*, in Advances in Neural Information Processing Systems 25, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds., Curran Associates, Red Hook, 2012, pp. 1097–1105.
[19] Y. LeCun, Y. Bengio, and G. Hinton, *Deep learning*, Nature, 521 (2015), pp. 436–444.
[20] J. Ling, A. Kurzawski, and J. Templeton, *Reynolds averaged turbulence modelling using deep neural networks with embedded invariance*, J. Fluid Mech., 807 (2016), pp. 155–166.
[21] Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang, *The expressive power of neural networks: A view from the width*, in Advances in Neural Information Processing Systems 30, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds., Curran Associates, Red Hook, 2017, pp. 6231–6239.
[22] R. Maulik and O. San, *A neural network approach for the blind deconvolution of turbulent flows*, J. Fluid Mech., 831 (2017), pp. 151–181.
[23] R. Maulik, O. San, A. Rasheed, and P. Vedula, *Subgrid modelling for two-dimensional turbulence using neural networks*, J. Fluid Mech., 858 (2019), pp. 122–144.
[24] R. D. Moser, S. W. Haering, and G. R. Yalla, *Statistical properties of subgrid-scale turbulence models*, Ann. Rev. Fluid Mech., 53 (2021), pp. 255–286.
[25] G. Novati, H. L. de Laroussilhe, and P. Koumoutsakos, *Automating turbulence modelling by multi-agent reinforcement learning*, Nature Mach. Intell., 3 (2021), pp. 87–96.
[26] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, *PyTorch: An imperative style, high-performance deep learning library*, in Advances in Neural Information Processing Systems 32, H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, eds., Curran Associates, Red Hook, 2019, pp. 8024–8035.
[27] S. B. Pope, *Turbulent flows*, 7th ed., Cambridge University Press, Cambridge, 2010.
[28] R. S. Rogallo, *Numerical experiments in homogeneous turbulence*, Tech. Report NASA-TM-81315, NASA Ames Research Center Moffett Field, CA, United States, September 1981.
[29] P. Sagaut and C. Cambon, *Homogeneous Turbulence Dynamics*, 2nd ed., Springer, Cham, 2018.
[30] F. Sarghini, G. de Felice, and S. Santini, *Neural networks based subgrid scale modeling in large eddy simulations*, Comput. Fluids, 32 (2003), pp. 97–108.
[31] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al., *Mastering the game of Go with deep neural networks and tree search*, Nature, 529 (2016), pp. 484–489.

[32] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, et al., *A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play*, Science, 362 (2018), pp. 1140–1144.

[33] J. Smagorinsky, *General circulation experiments with the primitive equations*, Mon. Weather Rev., 91 (1963), pp. 99–164.

[34] P. A. Srinivasan, L. Guastoni, H. Azizpour, P. Schlatter, and R. Vinuesa, *Predictions of turbulent shear flows using deep neural networks*, Phys. Rev. Fluids, 4 (2019), Art. 054603, 15 pages.

[35] B. D. Tracey, K. Duraisamy, and J. J. Alonso, *A machine learning strategy to assist turbulence model development*, in 53rd AIAA aerospace sciences meeting, January 2015, AIAA paper 2015–1287, AIAA, Reston, 2015, 22 pages.

[36] C. Xie, K. Li, C. Ma, and J. Wang, *Modeling subgrid-scale force and divergence of heat flux of compressible isotropic turbulence by artificial neural network*, Phys. Rev. Fluids, 4 (2019), Art. 104605, 43 pages.