INFLUENCE OF ADVERSARIAL TRAINING ON SUPER-RESOLUTION TURBULENCE MODELS

L. Nista[1], C. D. K. Schumann[1], M. Bode[1,2], T. Grenga[3], J. F. MacArt[4], A. Attili[5], and H. Pitsch[1]

[1]Institute for Combustion Technology, RWTH Aachen University, Aachen, 52056, Germany
[2]Jülich Supercomputing Centre (JSC), Forschungszentrum Jülich GmbH, Jülich, 52425, Germany
[3]Faculty of Engineering and Physical Sciences, University of Southampton, Southampton, SO17 1BJ, United Kingdom
[4]Aerospace and Mechanical Engineering, University of Notre Dame, Notre Dame, IN 46556, USA
[5]Institute for Multiscale Thermofluids, University of Edinburgh, Edinburgh, EH93FD, United Kingdom

ABSTRACT

Supervised super-resolution deep convolutional neural networks (CNNs) have gained significant attention for large eddy simulation subfilter-scale (SFS) modeling due to their ability to reconstruct statistically meaningful flow fields on fine meshes. Despite their popularity, CNNs lack the ability to accurately reconstruct high-frequency features and generalization performance on out-of-sample flows. Generative adversarial networks (GANs) are a potential alternative, allowing for both semi-supervised and fully unsupervised training, though they have not been thoroughly investigated as turbulence closures, and a comprehensive understanding of the discriminator's role has not been developed. This study assesses the effectiveness of GANs for *a priori* SFS stress modeling in forced homogeneous isotropic turbulence. It is found that GAN-based architectures outperform supervised CNN models for SFS reconstruction for in-sample cases. The reconstruction accuracy of both models decreases for out-of-sample data, though the GAN discriminator applied as a "feature extractor" narrows the model's solution space and enhances the generator's out-of-sample robustness. The extrapolation ability of the GAN-based model for higher-Reynolds-number flows is also demonstrated. This highlights the effectiveness of the GAN discriminator in optimizing robust and accurate SFS models for out-of-sample flows. Based on these findings, training with a discriminator is recommended before integrating super-resolution CNN closures into numerical solvers.

1 Introduction

Direct numerical simulation (DNS) accurately solves turbulent flows by resolving all relevant time and length scales but faces prohibitive computational costs for engineering applications. To address this challenge, the large eddy simulation (LES) approach solves spatially filtered equations, resulting in a significant cost reduction [1, 2]. The LES formulation distinguishes resolved (closed) and subfilter-scale (SFS; unclosed) fields. Accurate closure models for the SFS components are crucial for predicting the influence of the unresolved scales on the resolved scales [3].

Closure models must incorporate the unknown SFS fields into the evolution equations for the filtered fields $\overline{u}_i$ consistently with the mathematical and physical representation of the full-scale fields. Well-known closure models include the Smagorinsky model [4], dynamic models [5], and scale similarity models [6]. One prominent strategy involves employing a model form based on the deconvolution procedure [7]. This approach aims to approximate the unfiltered field $u_i^\bullet$ from the filtered field $\overline{u}_i$ by inverting the filtering operator $\mathcal{G}$,

$$u_i \approx u_i^\bullet = \mathcal{G}_l^{-1} * \overline{u}_i = \mathcal{G}_l^{-1} * \mathcal{G} * u_i \,, \tag{1}$$

where $\mathcal{G}_l^{-1}$ is an l^{th}-order approximate inverse of $\mathcal{G}$.

The effectiveness of the deconvolution approach depends on the accuracy of the approximate inverse operator, and the approach is only practical for invertible filters, i.e., for non-grid (non-projective) filters. Engineering LES calculations rely on the practice of "grid filtering," which, due to the Nyquist theorem, restricts deconvolution-based reconstruction to scales larger than $2\Delta^{\text{LES}}$, where Δ^{LES} is the LES mesh resolution [3]. Therefore, the full deconvolution process can be conceptually divided into two distinct sub-approaches, depicted in figure 1. The "soft" deconvolution approach involves reconstructing the unfiltered field to the grid-filter cutoff wavenumber, $k_c = \pi/\Delta^{\text{LES}}$, which does not account for the interactions between the resolved and subfilter scales. A secondary regularization approach, "hard" deconvolution, is necessary to account for the missing interactions.

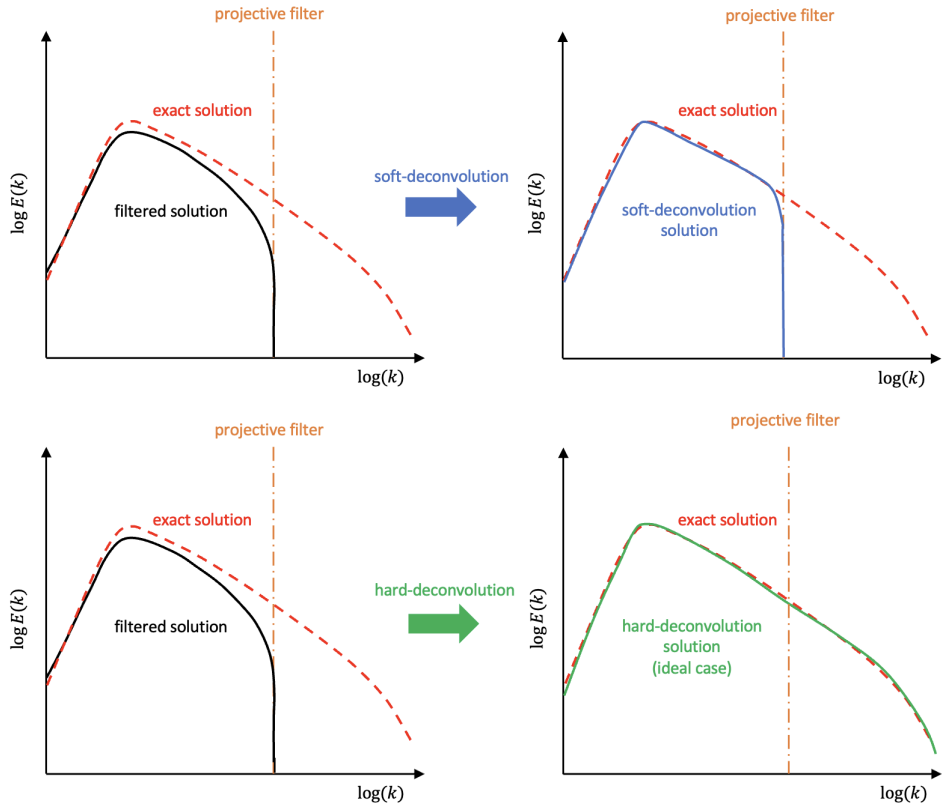

Figure 1: *Top*: the blue arrow indicates the "soft" deconvolution problem; *Bottom*: the green arrow indicates the ideal solution of the "hard" deconvolution problem [3].

Implementing deconvolution, therefore, requires two models. Various options exist to combine models for soft and hard deconvolution; examples are based on the scale-similarity hypothesis [6] or combine functional and structural models [8]. However, careful thought must be given to how functional and structural models are balanced [3].

Alternative methods have been proposed to solve the full deconvolution problem directly by reconstructing SFS fields on auxiliary grids; examples include the fractal interpolation method [9] and Terracol's multilevel algorithm [10]. Recently, data-driven super-resolution (SR) modeling has emerged as a promising method for SFS reconstruction on fine meshes with improved accuracy [11, 12, 13]. The SR approach aims to solve the hard deconvolution directly in a data-driven manner by using suitable neural network architectures to reconstruct higher-resolution (HR) fields from lower-resolution (LR) fields.

1.1 Data-driven super-resolution architectures for deconvolution methods

Deep convolutional neural networks (CNNs) have been applied for SR, with Fukami *et al.* [14, 15] pioneering their use for chaotic flows. They used a conventional super-resolution CNN to capture small-scale structures and reconstruct turbulent velocity and vorticity fields from low-resolution input data. By using a similar architecture with residual layers [16], Pant *et al.* [17] trained a deep CNN on forced homogeneous isotropic turbulence (HIT) to perform SR of LES-like fields, mitigating the tradeoff between resolution (fidelity) and computational complexity. Liu *et al.* [18] used a deep CNN model for SR reconstruction of 3D HIT, finding that training on time-series velocity fields improves

SR reconstruction capabilities. Recently, Zhou *et al.* [19] developed a turbulence volumetric super-resolution (TVSR) model based on CNNs, trained on various Reynolds numbers, that was robust and accurate for SR reconstructions at different Reynolds numbers, though only when coupled with an approximated deconvolution method (ADM).

While SR architectures have been shown to be more accurate than algebraic models for in-sample predictions [19, 20], the current understanding of SR mainly involves results obtained from supervised training of CNN models. These models require large amounts of labeled data, which can be impractical to obtain (often from DNS). Supervised SR-CNN-based architectures do not generalize well to out-of-sample flow conditions or domains, for they are optimized for specific datasets and objective functions, hence they remain largely untested for out-of-sample inputs. Furthermore, supervised CNN architectures do not accurately predict high-resolution details, resulting in a loss of high-wavenumber features and thus blurry super-resolved fields. This is a significant limitation for applications deconvolution methods that require accurate predictions of high-frequency detail.

To address the drawbacks of the supervised CNNs training strategy for SR, generative adversarial networks (GANs) have been employed [21] which, unlike CNNs, do not rely solely on user-defined objective functions for training. Instead, GANs' super-resolving generator networks, based on CNNs, are coupled with discriminator networks that compete in semi-supervised learning. In GAN-based SR, the generator produces higher-resolution versions of low-resolution fields (e.g., filtered DNS or LES), while the discriminator network attempts to differentiate between the generated high-resolution field and the real, "ground truth" high-resolution field (e.g., the DNS itself). During training, the generator learns to produce samples that are indistinguishable from genuine high-resolution data. The discriminator in turn learns to judge the authenticity of the samples. As a result, both networks improve each other during the adversarial training process.

Deng *et al.* [22] compared super-resolution GAN (SRGAN) [23] and enhanced SRGAN (ESRGAN) [24] approaches, initially developed for image reconstruction, for the reconstruction of the flow around tandem cylinders by upscaling two-dimensional turbulence. The ESRGAN architecture outperformed the SRGAN in mean-flow metrics and fluctuation distributions. Through the use of GAN architectures with and without physical loss functions (based on mass and momentum conservation), Lee *et al.* [25] studied the prediction of flow around a cylinder in laminar conditions. Perhaps one should also note that this approach barely differs from DNS and is thus, by default, computationally prohibitive. In reconstructing temporal data from a large time-step interval, only the physics-based loss function adequately constrained the solution space for accurate prediction of the resolved flow motion.

The capability to accurately approximate the hard deconvolution operation has made GAN-based networks attractive for SFS closure modeling. Bode *et al.* [26, 27] introduced a physics-informed ESRGAN (PIESRGAN) for SFS turbulence reconstruction incorporating a loss function based on the continuity-equation residual alone. Although their model was only trained on HIT data, it was able to improve the predictive accuracy of scalar mixing in a reacting jet. Nista *et al.* [28, 29] then investigated the generalization capability of a similar GAN architecture by evaluating the model's extrapolability to higher and lower Reynolds numbers than those used for training. They found that the ratio between the LES filter width and the Kolmogorov scale must be preserved for adequate generalization (i.e., a fixed SR upsampling window). Grenga *et al.* [30] investigated the ability of similar GAN models to recognize and reconstruct gradient and counter-gradient transport in low- and high-Karlovitz-number combustion regimes [31]. In this case, only a GAN trained for both datasets adequately reconstructed the subfilter scales for both combustion regimes. However, the underlying reasons for this success are not yet fully understood and need further investigation. On one hand, adversarial training might enhance the generalization capability of a given network. On the other hand, possible drawbacks include training instability and mode collapse, the latter of which can hinder convergence of the generator [32]. Given the computational expense of training GAN architectures, a critical evaluation of the need for GAN architectures, and the influence of the discriminator on generalizability, is needed. This can provide valuable insights into the efficacy of GAN architectures for universal SFS modeling.

This study investigates and quantifies the role of the GAN discriminator in SR-based SFS models while considering the benefits and limitations of the GAN approach. To evaluate the effectiveness of the adversarial training, a direct comparison between a supervised deep CNN model and a semi-supervised/unsupervised GAN architecture (PIESRGAN) is performed with physics-inspired loss functions in both. For a fair comparison, the same CNN network is used as the generator for PIESRGAN along with the same loss functions and training data. Hence, the sole difference between the two is in the training strategy. In addition, as most approaches have focused on in-sample reconstruction under a narrowly confined set of conditions, we explore the generalizability of both models to out-of-sample input fields obtained for different filter sizes and filter kernels. The performance of both models is also evaluated at higher Reynolds numbers.

2 Datasets and preprocessing

Three forced HIT DNS datasets at Taylor-microscale Reynolds numbers $Re_\lambda = 90$, $Re_\lambda = 130$, and $Re_\lambda = 350$ are considered. These are computed using the CIAO code, which uses second-order central differences, staggered meshes, and second-order implicit time advancement to solve the incompressible Navier–Stokes equations [33]. The calculations have mesh sizes 256 $\times$ 256 $\times$ 256 points for the low Re, 512 $\times$ 512 $\times$ 512 points for intermediate Re, and 1024 $\times$ 1024 $\times$ 1024 points for the high Re. The linear forcing $\mathbf{f} = A\,\mathbf{u}$ proposed by Lundgren *et al.* [34] is applied, where A is a forcing parameter inversely proportional to the eddy turnover time. Table 1 reports the simulation parameters for the three datasets. SR training used only the lowest Re DNS (Re90), with the higher Re DNSs (Re130 and Re350) being reserved for testing.

After the lowest Re simulation reached a statistically stationary state, 130 snapshots of the 3D velocity vector (u_i = (U, V, W)) were extracted for training. To obtain low-resolution data, all available snapshots of the instantaneous velocity were filtered using explicit filter kernels of width $\Delta = 4\,dx$, ensuring that 14% of the turbulent kinetic energy (TKE) is at the subfilter-scale, which is reasonable for LES [35]. The filtered data, therefore, have consistent TKE with accepted standards for well-resolved LES. The training, validation, and testing datasets were composed of high-resolution data, i.e., DNS data, and the corresponding low-resolution data, i.e., filtered DNS (F-DNS) data. The spatial box, Gaussian, and spectrally sharp filter kernels are

$$\mathcal{G}_{\text{box}}(\boldsymbol{x}) = \begin{cases} \frac{1}{\Delta} & \text{if } |\boldsymbol{x}| \leq \frac{\Delta}{2} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

$$\mathcal{G}_{\text{Gaussian}}(\boldsymbol{x}) = \left(\frac{6}{\pi\Delta^2}\right)^{\frac{1}{2}} \exp\left(-\frac{6(\boldsymbol{x})^2}{\Delta^2}\right), \tag{3}$$

$$\mathcal{G}_{\text{spectral}}(\boldsymbol{x}) = \frac{\sin\left(\pi(\boldsymbol{x})/\Delta\right)}{\pi(\boldsymbol{x})/\Delta}, \tag{4}$$

and the low-resolution F-DNS fields are obtained by a discrete downsampling operation, applied independently of the filter kernel, defined as

$$\mathbf{f}_{\text{downsampling}} : \mathbb{R}^{\Omega} \rightarrow \mathbb{R}^{\Omega/\Delta},\ \bar{\mathbf{u}}_{\text{F-DNS}} = \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{N} \mathcal{G} * \mathbf{u}(\Delta i, \Delta j, \Delta k)\,, \tag{5}$$

where N is the number of mesh points of the high-resolution field and Ω is the corresponding three-dimensional field. The latter operation ensures that the F-DNS fields have the same dimensionality as the corresponding LES fields. A comparison of filter kernels (box, Gaussian, and spectrally sharp) is given in Section 5.1.

Case	N	Re_λ	Re_t	L_x/L_{11}	dx/η	k_{max}	Train	Test
Re90	256^3	90	920	5.26	1.98	128	•	•
Re130	512^3	130	3320	5.26	1.86	256		•
Re350	1024^3	350	16182	5.26	1.97	512		•

Table 1: Simulation parameters for training and testing datasets. N is the number of mesh points, Re_t is the turbulent Reynolds number, L_x/L_{11} is the number of integral scales within the computational domain, dx/η is the mesh resolution relative to the Kolmogorov microscale η, and k_{max} is the largest wavenumber represented. "Train" and "test" flags indicate which data were used in model training and testing, respectively.

Loading the entire training dataset is impractical given memory limitations. To alleviate this, we extract sub-domains randomly from each snapshot of the full domain. The first 110 snapshots, were used for training, while the last 20 frames, equally divided, were used for validation and testing (training/validation ratio = 90%/10%). The validation was then performed on 19 in-sample fields not seen during the training, and only the last snapshot was employed for *a priori* testing. Both networks, TSResNet and PIESRGAN (introduced in section 3) were trained using sub-boxes of the original computational domain with a size of $64 \times 64 \times 64$ and $16 \times 16 \times 16$ for high- and low-resolution fields, respectively. The size of these sub-boxes was found to be a good compromise between memory requirements and the integral length scale of the flow. Using those settings, 6400 sub-boxes were employed for the training. The velocity components of high-resolution data and low-resolution data used for training and testing were normalized by the global maximum and minimum of the DNS (i.e., $(U_{F-DNS}, V_{F-DNS}, W_{F-DNS}) \in [0,1]$ and $(U_{DNS}, V_{DNS}, W_{DNS}) \in [0,1]$) to improve the network's performance [36].

The large dataset and deep convolutional frameworks (implemented in TensorFlow, [37]), required training parallelization; this was done using the Horovod library [38]. Calculations were performed on the Jülich DEEP-EST cluster (DEEP-DAM partition) using four nodes, each with one NVIDIA V100 32GB GPU. This gave an overall speed-up close to a factor of four versus single-GPU training. A mini-batch of 8 sub-boxes per GPU and the ADAM optimizer, with an initial learning rate of 10^{-4}, were chosen due to memory constraints and previous evidence of the success of this combination [17, 25, 39]. "Mixed-precision" training is common in machine learning for its low computational cost [40] but can be detrimental to predictive accuracy in scientific computing. We use single-precision arithmetic for both weight definition and loss computations, following the results of Hrycej *et al.* [41]. Details are provided in appendix A.

For unsupervised training, LES calculations of forced HIT are conducted using the same code for DNS at coarser grid resolutions (equivalent to the discrete downsampling factor applied to the DNS mesh). The unresolved scales are closed using the dynamic Smagorinsky model [5, 42] (equation 11), or their influence is neglected ("implicitly modeled" LES). To extrapolate a given model to higher Re (utilizing the Re130 and Re350 datasets for out-of-sample analyses), the filtered input field is rescaled to match the Δ/η ratio used for training. This rescaling procedure guarantees a certain level of generalizability [28].

3 Neural network architectures and training strategy

Our networks are based on the architecture of Bode *et al.* [26, 27, 43]. Differing from those previous works, our generator network uses additional up-/downsampling layers and dense layers, depicted in figure 2; this is inspired by the original ESRGAN architecture [24] and adapted for small-scale turbulence reconstruction [29]. Along with the definition of the two architectures, the training approaches are described.

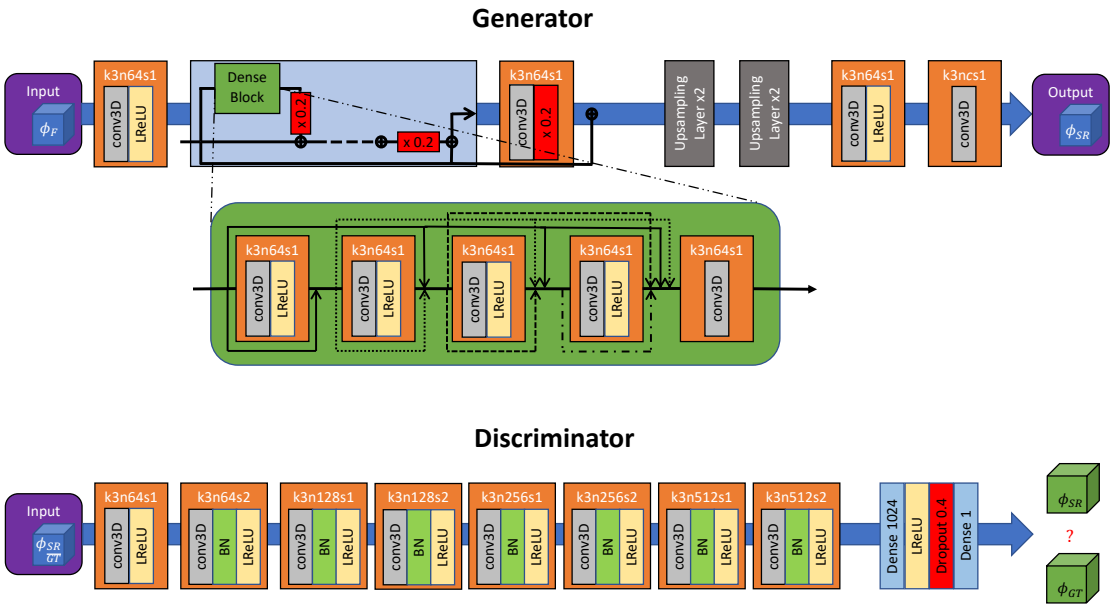

Figure 2: The structure of the generator (above) was employed for both networks. The discriminator (below) was only employed for the PIESRGAN architecture. ϕ_F indicates the filtered input field, ϕ_{SR} is the super-resolved field, and ϕ_{GT} refers to the ground-truth (DNS) field. Moreover in each convolutional block, *k* is the kernel size, *n* is the number of filter maps, and *s* is the strides along each spatial dimension of a convolutional layer.

TSResNet

The TSResNet (Turbulent Super-Resolution Residual Network) is a supervised CNN-type model. It uses only the generator shown in figure 2 without the discriminator. The TSResNet is adapted from the original SRResNet [44], which

can capture small-scale features when skip-connections are included. The generator relies on three-dimensional convolutional layers with Leaky Rectified Linear Units (LReLUs) as activation functions, as they have higher computational efficiency versus standard rectified linear units (ReLUs) [37]. The Residual-in-Residual Dense Blocks (RRDBs) [24] contain fundamental architecture components, such as skip-connections and dense blocks (three in the present work), that enable the generation of super-resolved data through a very deep network. This is expected to facilitate learning complex transformations. Two upsampling layers increase the input spatial resolution by a factor of 4 in each spatial dimension. Each one doubles the dimensions of the input by nearest-neighbor interpolation (replicating adjacent grid points), followed by a convolution layer to improve the approximation. The upsampling branch (the gray section in figure 2) is later modified in section 5.2, where additional upsampling layers are adopted. The total number of trainable parameters in the TSResNet is approximately 18 million.

The perceptual loss originally proposed by Wang et al. [24] for image reconstruction has been found to be less suitable for turbulence modeling [20] and is thus replaced with constraints derived from the continuity equation. The loss function used for the TSResNet training process ($\mathcal{L}_{\text{CNN}}$) is a combination of pixel loss (L_{pixel}), pixel gradient loss (L_{gradient}) and continuity loss ($L_{\text{continuity}}$),

$$\begin{aligned} \mathcal{L}_{\text{CNN}} &= \beta_1 L_{\text{pixel}} + \beta_2 L_{\text{gradient}} + \beta_3 L_{\text{continuity}} \\ L_{\text{pixel}} &= MSE(\phi_{\text{SR}}, \phi_{\text{DNS}}) \\ L_{\text{gradient}} &= MSE(\nabla\phi_{\text{SR}}, \nabla\phi_{\text{DNS}}) \\ L_{\text{continuity}} &= MSE(\nabla \cdot \phi_{\text{SR}}, 0), \end{aligned} \tag{6}$$

where the coefficients $\beta = [0.89, 0.06, 0.05]$ were previously selected by Bode *et al.* [26] though hyperparameter tuning. ϕ_F is the filtered input field, ϕ_{SR} is the super-resolved field, and ϕ_{DNS} is the DNS field. The mean-squared error (MSE) is computed between the reconstructed and ground-truth fields, as indicated in equation 6. The MSE operator is applied separately to all elements when tensor quantities are considered. The network was initialized with random weights to ensure unbiased predictions. Training occurs for a sufficient number of epochs after which the loss function and statistics computed on the reconstructed field do not change substantially.

PIESRGAN

In contrast, the PIESRGAN model uses the full GAN architecture, i.e., generator and discriminator in adversarial training. The generator has the same architecture as the TSResNet described previously. The discriminator (figure 2) is a deep deconvolutional architecture of fully connected layers with binary classification output, differing from the original ESRGAN discriminator by the introduction of a dropout layer to prevent over-fitting. The total number of trainable parameters of the discriminator is approximately 19 million.

The generator's loss function is largely unchanged compared to the TSResNet but now includes the contribution of the discriminator loss ($L_{\text{adversarial}}$, [45]). The generator loss function is $\mathcal{L}_{\text{GAN}} = \beta_1 L_{\text{pixel}} + \beta_2 L_{\text{gradient}} + \beta_3 L_{\text{continuity}} + \beta_4 L_{\text{adversarial}}$, where

$$\begin{aligned} L_{\text{adversarial}} = -\mathbb{E}[\log((D(G(\phi_{\text{SR}})) - \mathbb{E}[D(\phi_{\text{DNS}})]))] - \\ \mathbb{E}[\log(1 - (D(\phi_{\text{DNS}}) - \mathbb{E}[D(G(\phi_{\text{SR}}))]))], \end{aligned} \tag{7}$$

$\mathbb{E}[\cdot]$ is the expectation operator, and $D(\phi_{\text{DNS}})$ and $G(\phi_{\text{SR}})$ are the discriminator and generator outputs. The first term of the adversarial loss function encourages the discriminator to correctly classify HR fields as real, while the second term encourages the generator to produce SR fields that can fool the discriminator into classifying them as HR. If ϕ_{DNS} is not provided during the training, e.g., unsupervised training (described below), $D(\phi_{\text{DNS}})$ is equal to zero. The weights $\beta = [0.88994, 0.06, 0.05, 6 \cdot 10^{-5}]$ were chosen through hyperparameter tuning, such that the absolute value of each term in $\mathcal{L}_{\text{GAN}}$ is of the same order. The loss function, used to train the discriminator in a supervised manner, is given by

$$\begin{aligned} \mathcal{L}_{\text{DISC}} = \mathbb{E}[\log((D(\phi_{\text{DNS}}) - \mathbb{E}[D(G(\phi_{\text{SR}}))]))] + \\ \mathbb{E}[\log(1 - (D(G(\phi_{\text{SR}})) - \mathbb{E}[D(\phi_{\text{DNS}})]))], \end{aligned} \tag{8}$$

which is based on the relativistic average GAN loss function proposed by Jolicoeur *et al.* [45].

The training of the GAN architecture is challenging, as the generator and discriminator networks are trained to compete against each other. Finding a convergence point is one of the main challenges of GAN training, as the training might suffer from oscillations and destabilization in the model's trainable parameters [21]. To cope with these, the generator is pre-trained in a fully-supervised manner (like the TSResNet) by using the original generator loss function proposed in equation 6. Subsequently, the pre-trained generator is used to initialize the GAN training. Recent findings demonstrated that the choice of the initial learning rate for both networks is essential for the local convergence of the GAN training, and recommendations from recent literature have been adopted [32, 29].

Training strategy

Table 2 lists the architecture, training strategy, and training/testing fields for the specific trained models used in subsequent sections. The TSResNet models were trained for roughly 200 epochs, after which no additional reconstruction improvement was observed, using the *Re90* dataset and (6) as the loss function. Statistical convergence required 37 wall-time hours using four NVIDIA Tesla V100 32GB GPUs. The trained TSResNet model parameters were used to initialize the GAN generators. No GAN discriminator pre-training was applied, as this step has recently been demonstrated to have no influence [29]. The adversarial training required an additional 11 wall-time hours using the same hardware, i.e., roughly $30\,\%$ more than the TSResNet training time, over an additional 40 epochs. To counteract the influence of the additional training time performed during the GAN training, the TSResNet models were trained for an additional 40 epochs. However, this additional training time did not impact the final reconstructed field and can be considered superfluous.

Model	Training strategy	Input training – testing fields: Filter kernels ($\overline{\Delta}/\Delta_{\text{DNS}}$)	Section
	Influence of adversarial training on in-sample predictions		
TSResNet (spectral kernel)	supervised	S (4) – S (4)	§4
TSResNet (box kernel)	supervised	B (4) – B (4)	§4
PIESRGAN (spectral kernel)	semi-supervised	S (4) – S (4)	§4
PIESRGAN (box kernel)	semi-supervised	B (4) – B (4)	§4
	Influence of adversarial training on out-of-sample filters		
PIESRGAN (multiple filter kernel)	semi-supervised	B $\wedge$ S (4) – G (4)	§5.1
PIESRGAN (partially unsupervised)	semi-supervised $\&$ unsupervised	B $\wedge$ S $\wedge$ LES (4) – G (4)	§5.1
TSResNet (fixed upsampling)	supervised	B (4) – B (8)	§5.2
PIESRGAN (fixed upsampling)	semi-supervised	B (4) – B (8)	§5.2
PIESRGAN ($x\#$)	semi-supervised $\&$ dynamic upsampling	B ($2 \wedge 4 \wedge 8$) – B ($\#$)	§5.2
	Influence of adversarial training on out-of-sample predictions (higher Re numbers)		
TSResNet	supervised (Re90)	B (4) – B (4)	§6
PIESRGAN	semi-supervised (Re90)	B (4) – B (4)	§6

Table 2: List of model configurations, training strategies, and training/testing filter kernels and sizes. The filter kernels "B," "G," and "S" abbreviate box, Gaussian, and spectrally-sharp kernels. The $\wedge$ indicates that the combined set of the input fields is employed in the training, while the $\#$ indicates the variable upsampling factor.

4 Effects of adversarial training on in-sample SFS reconstruction

Both models are tested on in-sample data, statistically matched with the training data, to assess their SFS reconstruction capability; this evaluates the models' performance under ideal conditions. Figures 3 and 4 compare TKE spectra and the probability density function (PDF) of normalized velocity gradients of the filtered DNS (F-DNS), full-mesh (unfiltered) DNS, TSResNet upsampled fields, and PIESRGAN upsampled fields. The SR models aim to upsample the F-DNS to match DNS fields. The models are trained and evaluated for the Re90 case using either the box filter (figure 3) or spectrally sharp filter (figure 4). The primary difference in the training data is the box filter's artificial attenuation of the near-filter-scale F-DNS fields, which the spectrally sharp filter avoids. The TSResNet recovers the TKE of the resolved scales ($k \gtrsim 32$) but fails to correctly predict the SFS energy distribution for either filter. Conversely, the PIESRGAN is more accurate, deviating from the DNS only for $k \gtrsim 122$ and energy values $E(k) \lesssim 10^{-7}$ for both filters. This behavior is quantitatively confirmed by the PDF of the normalized velocity gradient, where the TSResNet tends to underpredict the large gradients, while the PIESRGAN marginally deviates from the DNS solution. It is worth noting that the TSResNet and the PIESRGAN generator use the same network architecture and the same basic loss function. The PIESRGAN's more-accurate SFS reconstruction is therefore due to its training, rather than network depth or loss-function differences.

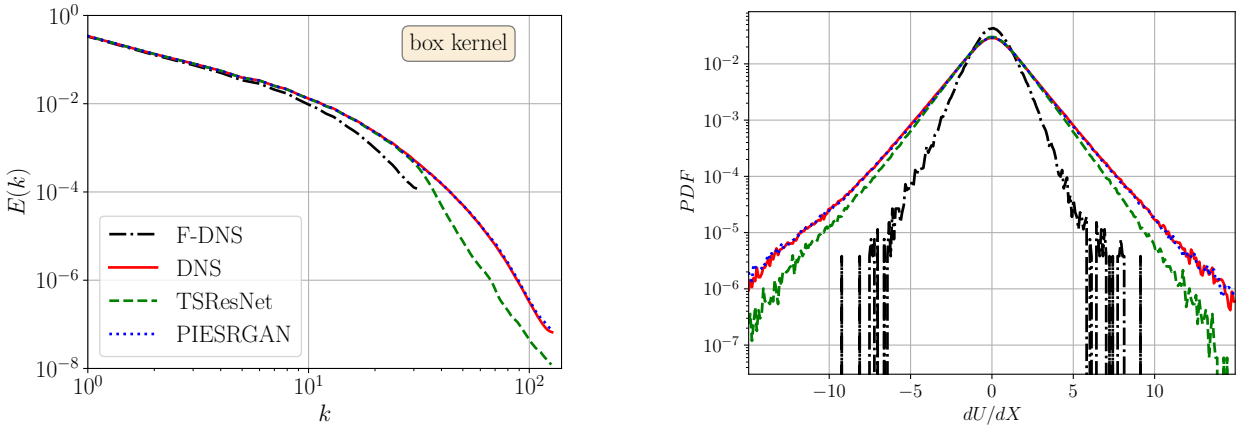

Figure 3: In-sample (Re90) TKE spectra (left) and PDF of the normalized velocity gradient (right) for box-filtered training/testing fields.

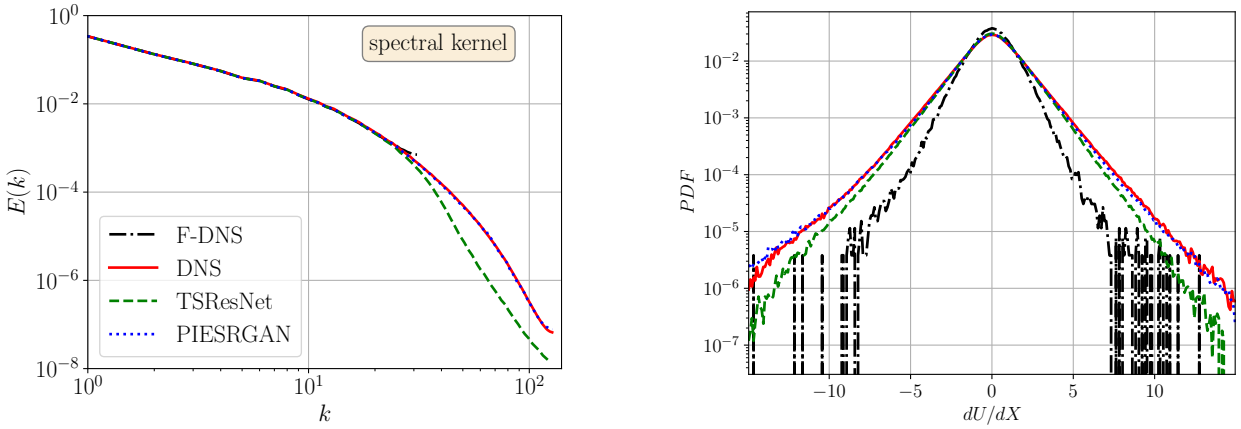

Figure 4: In-sample (Re90) TKE spectra (left) and PDF of the normalized velocity gradient (right) for spectrally-sharp filtered training/testing fields.

In terms of deconvolution operations, the TSResNet model appears to be equivalent to the soft deconvolution, as it improves the resolved scales but does not account for the interaction between resolved and subfilter scales. For *a posteriori* deployment, the TSResNet model would require another SFS model such as an eddy-viscosity model. Conversely, the PIESRGAN architecture can be interpreted as a generalized deconvolution-based model, for it does

not require a second regularization to model interactions with the SFS modes. The PIESRGAN is therefore a more general model, able to perform soft and hard deconvolution simultaneously.

Probability density functions (PDFs) of the normalized velocity increments $\delta_X U$ for different separation distances proportional to the Kolmogorov microscale η were considered to characterize turbulent structures at different scales, similar to Attili *et al.* [46]. Figure 5 compares the DNS and reconstructed fields for velocity increments of 2η (solid lines), 4η (dotted lines), 16η (dashed lines), 64η (dash-dot lines). For small increments, the TSResNet captures only the most probable features well but underpredicts at the tails. For the largest increment (64η), the gap with the ground truth is less evident. These TSResNet reconstructions are perceptually smoother and blurred compared to the DNS fields. Conversely, the PIESRGAN predictions almost exactly overlap with the DNS for all increments considered. This is due to the discriminator's nonlinear feedback to the generator (due to the discriminator's nonlinear activation functions), which boosts the generator's accuracy for the subfilter scales. This means that the GAN can reconstruct complex, nonlinear relationships between low-resolution inputs and the reconstructed fields. The adversarial training, therefore, improves the *a priori* SFS reconstruction capabilities compared to supervised learning, even with the same generator architecture. For *a posteriori* predictions, only the GAN generator is needed, so the TSResNet and GAN have identical prediction costs.

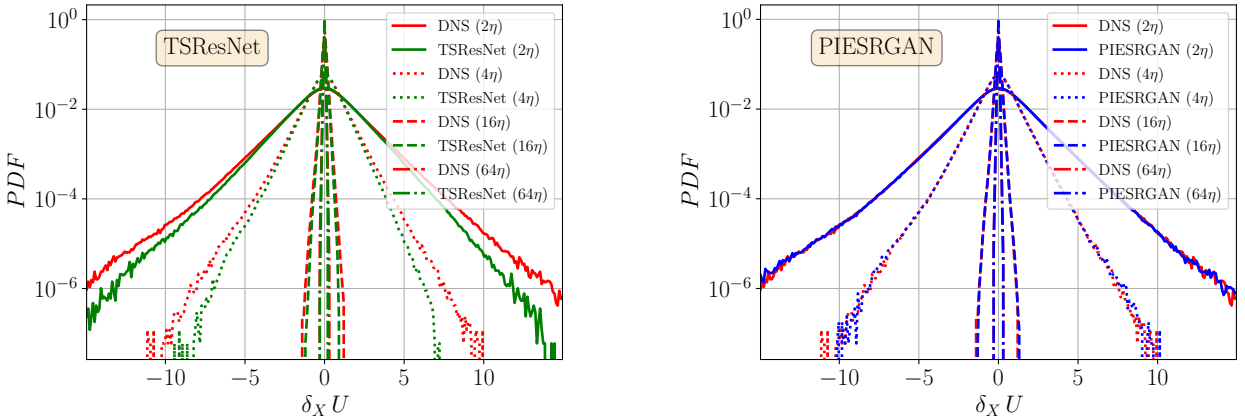

Figure 5: PDFs of the normalized velocity increments for different separation distances for box-filtered DNS fields, TSResNet reconstructions (left), and PIESRGAN reconstructions (right).

One goal of super-resolution is to develop SFS closures for LES; it is, therefore, important to consider the accuracy of their *a priori* SFS reconstruction. The LES-filtered momentum equation,

$$\frac{\partial \overline{u}_j}{\partial t} + \overline{u}_i \frac{\partial \overline{u}_j}{\partial x_i} = -\frac{1}{\rho}\frac{\partial \overline{p}}{\partial x_j} + \nu \frac{\partial^2 \overline{u}_j}{\partial x_i \partial x_i} - \frac{\partial \tau_{ij}^{\text{SFS}}}{\partial x_i}, \tag{9}$$

contains the anisotropic residual-stress tensor τ_{ij}^{SFS},

$$\tau_{ij}^{\text{SFS}} \equiv \overline{u_i u_j} - \overline{u}_i\,\overline{u}_j - \frac{2}{3}k^{\text{SFS}}\delta_{ij}, \tag{10}$$

where u_i are unfiltered velocity components obtained from either the DNS or the SR fields and $k^{\text{SFS}} \equiv (\overline{u_i u_i} - \overline{u}_i \overline{u}_i)/2$ is the kinetic energy of the subfilter scales [35]. In equation 10, the $\bar{\cdot}$ operator indicates filtering (either box or spectrally-sharp filters) and downsampling proportional to the upsampling factor (in the present case, the factor is equal to four). Comparisons are made to the widely used dynamic Smagorinsky model [5, 42],

$$\tau_{ij}^{\text{SFS}} = -2c_s\overline{\Delta}^2\,|\overline{S}|\overline{S}_{ij}, \text{ with } |\overline{S}| = (2\overline{S}_{ij}\overline{S}_{ij})^{1/2} \tag{11}$$

where $\overline{S}_{ij}$ denotes the filtered strain-rate tensor, $\overline{\Delta}$ is the filter width, and c_s is the dynamically computed Smagorinsky constant, averaged along the homogeneous directions [42].

Figure 6 shows joint PDFs (jPDFs) of the box-filtered DNS τ_{12}^{SFS} and the dynamic-Smagorinsky (left), TSResNet-reconstructed (center), and PIESRGAN-reconstructed (right) τ_{12}^{SFS}. The PIESRGAN-reconstructed field is statistically more similar to the filtered DNS than the TSResNet-reconstructed and dynamic-Smagorinsky-modeled fields, in order of decreasing fidelity. The PIESRGAN improves the root mean squared error (RMSE) by 64 % compared to the TSResNet. In general, each component of the PIESRGAN-reconstructed τ_{ij}^{SFS} has an average pointwise correlation,

computed from the jPDF, with the DNS exceeding $90\,\%$. Similar results are obtained for spectrally filtered input fields. It is clear that adversarial training improves the *a priori* correlation of SR fields with ground-truth data for in-sample predictions.

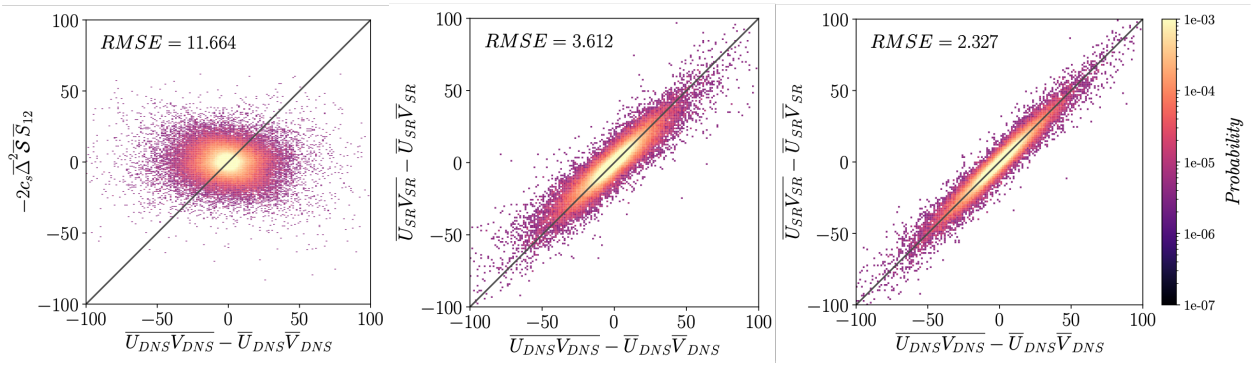

Figure 6: Joint PDFs of the box-filtered DNS τ_{12}^{SFS} with the dynamic Smagorinsky-modeled field (left), TSResNet reconstruction (center), and PIESRGAN reconstruction (right).

5 Accuracy of SR models for out-of-sample filters

Recent work has highlighted practical limitations of data-driven deconvolution methods arising primarily when the application differs substantially from the training data [21, 47, 48]. In the previous section, the PIESRGAN successfully reconstructed in-sample filtered fields ϕ_F and approximately solved the full deconvolution problem. Hence, the model is essentially capable of approximating the inverse operator $\mathcal{G}^{-1}$ using analytical filtering and discrete downsampling. Models trained on F-DNS data, for which the corresponding full-resolution fields are known, must also accurately upsample real-world LES fields. However, due to the interactions between grid spacing, numerical accuracy, and modeling assumptions, the implicit filter applied to practical LES calculations is unknown in terms of its effective kernel and filter width. To understand the influence of these, we now consider out-of-sample filter kernels and filter widths separately.

5.1 Different filter kernels for training and testing

The trained SR models are now tested for input fields obtained by filtering the DNS with out-of-sample (i.e., unused in training) filter kernels. This emulates the *a posteriori* application of the trained models to practical LES calculations, for which the implicit filter operation is unknown.

Figure 7 compares the TKE spectra and the PDF of normalized velocity gradients for the networks that were trained individually on box-filtered or spectrally-sharp filtered fields exclusively and applied to Gaussian-filtered input data. The exclusively box-filter-trained SR models consistently overestimate the SFS kinetic energy, which results in inaccurate predictions at the high wavenumbers and overpredicted energy at lower wavenumbers than the filter cutoff. This overestimation can be especially problematic in *a posteriori* deployment, as it can lead to error accumulation and insufficient dissipation, resulting in unstable and inaccurate solutions. Conversely, for models exclusively trained on spectrally sharp filtered data, the opposite behavior for Gaussian-filtered reconstruction is observed: both models underpredict the resolved and SFS kinetic energy, with the PIESRGAN model being slightly more accurate than the TSResNet model. Clearly, for unknown filter kernels (absent from the training data), as for grid-filtered LES, the performance of both SR models drops drastically.

One possible approach to address the reduced out-of-sample filter invertibility is to combine an SR architecture with an algebraic approximate deconvolution method (ADM) [7]. This would first use the ADM to recover the resolved velocity fluctuations (essentially a soft deconvolution), then a PIESRGAN trained for spectrally-sharp filtered fields would extrapolate the subfilter scales (hard deconvolution). In this arrangement, the GAN would act as a second regularizer ([3]), though it is important to note that the ADM-deconvolved field would still differ from an ideal, spectrally-sharp filtered field, especially near the cutoff scale [19, 49]. Thus, the performance of the spectral-trained GAN could suffer from its tendency to invert a known filter. A combination of an SR data-driven model and an algebraic ADM could partially minimize the GAN's over/underestimation, but it would likely not completely solve the issue.

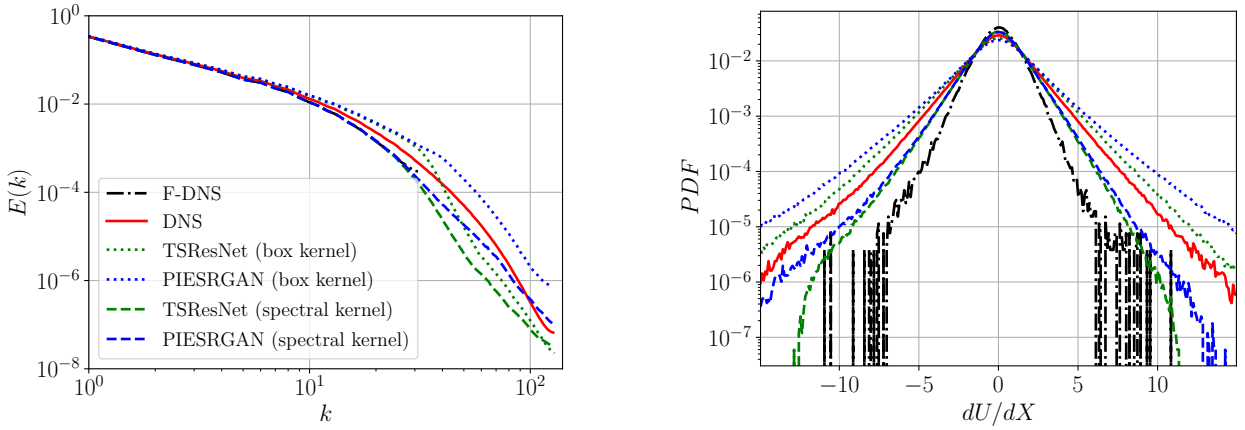

Figure 7: TKE spectra (left) and PDF of the normalized velocity gradient (right) for models trained either for box- or spectrally sharp-filtered data and tested on Gaussian-filtered data.

Another potential solution leverages the GAN's flexibility to train both in a semi-supervised and unsupervised mode, relying on the underlying "structure" of the data for patterns. Initially, both models were trained on a mixture of box-filtered and spectrally-sharp filtered data in an otherwise consistent manner with the training strategy outlined previously. This approach is henceforth referred to as *multiple-kernel-trained*. The discriminator is then trained to distinguish between generated high-resolution fields, upsampled from either box-filtered or spectrally-sharp filtered DNS fields, and the corresponding ground-truth fields. Validating for in-sample F-DNS fields (i.e., obtained with the same filter kernel used for training), the reconstruction performance is comparable to that obtained in section 4. Similar to the previous analysis, the PIESRGAN gives better reconstruction capability than the TSResNet; hence we omit the TKE spectra for brevity. Both models are capable of recognizing and learning both filtering operations that were applied to generate low-resolution training data, suggesting that this ability is not exclusively derived from adversarial training.

When testing for out-of-sample Gaussian-filtered input data (not included in the training data), the multiple-kernel-trained PIESRGAN model again suffers a performance drop. Figure 8 plots centerline slices of the normalized velocity magnitude and its absolute error obtained for the PIESRGAN using multiple-filter-kernel training. It is evident that the velocity field reconstructed by the model exhibits distortions compared to the ground-truth data. Notably, the error is especially pronounced for small-scale structures. Figure 9 shows TKE spectra and PDFs of normalized velocity gradients for the multiple-kernel-trained PIESRGAN model, showing that the reconstructed field is in better agreement with the DNS than the single-filter-trained model, though artifacts remain in the high-frequency subfilter scales. The velocity-gradient PDF shows similar trends for the multiple-kernel-trained PIESRGAN, with consistent overprediction of the small-scale shear strain. Training for different filter kernels can mitigate filter-inversion inadequacies, but does not alleviate them altogether.

While the discriminator does not directly influence the generator's ability to distinguish between filter kernels, adversarial training can still improve the accuracy of the reconstructed fields. This is an advantage of the GAN-based architecture. Inspired by Bode *et al.* [26], though differing in purpose and methodology, we now test using both labeled (corresponding DNS and F-DNS fields) and unlabeled data (computed LES fields, either with no SFS model or the dynamic Smagorinsky model, as described in section 3) in the training process. Labeled data helps the discriminator differentiate between real and fake samples, while unlabeled data improves the generator's ability to upsample realistic fields. The discriminator is trained in a supervised manner using only generated and ground-truth fields, and its weights are not updated in the subsequent LES-training step, while the generator continues to be trained on randomly sampled LES data. The resulting super-resolved LES fields are then evaluated by the trained discriminator. Since the discriminator is not trained further, the generator loss was reduced to

$$\mathcal{L}_{\mathrm{GAN}_{\mathrm{unsupervised}}} = \beta_3 \, L_{\mathrm{continuity}} + \beta_4 \, L_{\mathrm{adversarial}}, \tag{12}$$

where $\beta_3 = 0.05$ and $\beta_4 = 6 \cdot 10^{-5}$. This is referred to as the semi-supervised approach.

The right column of Figure 8 demonstrates that the PIESRGAN model, trained in a partially unsupervised manner, precisely reconstructs small-scale structures absent from the (out-of-sample) input data, with significantly lower normalized absolute error. Thus, the use of partially unsupervised training significantly reduces the overprediction of small features. Figure 9 illustrates decreased overprediction of the SFS kinetic energy and better reconstruction of the

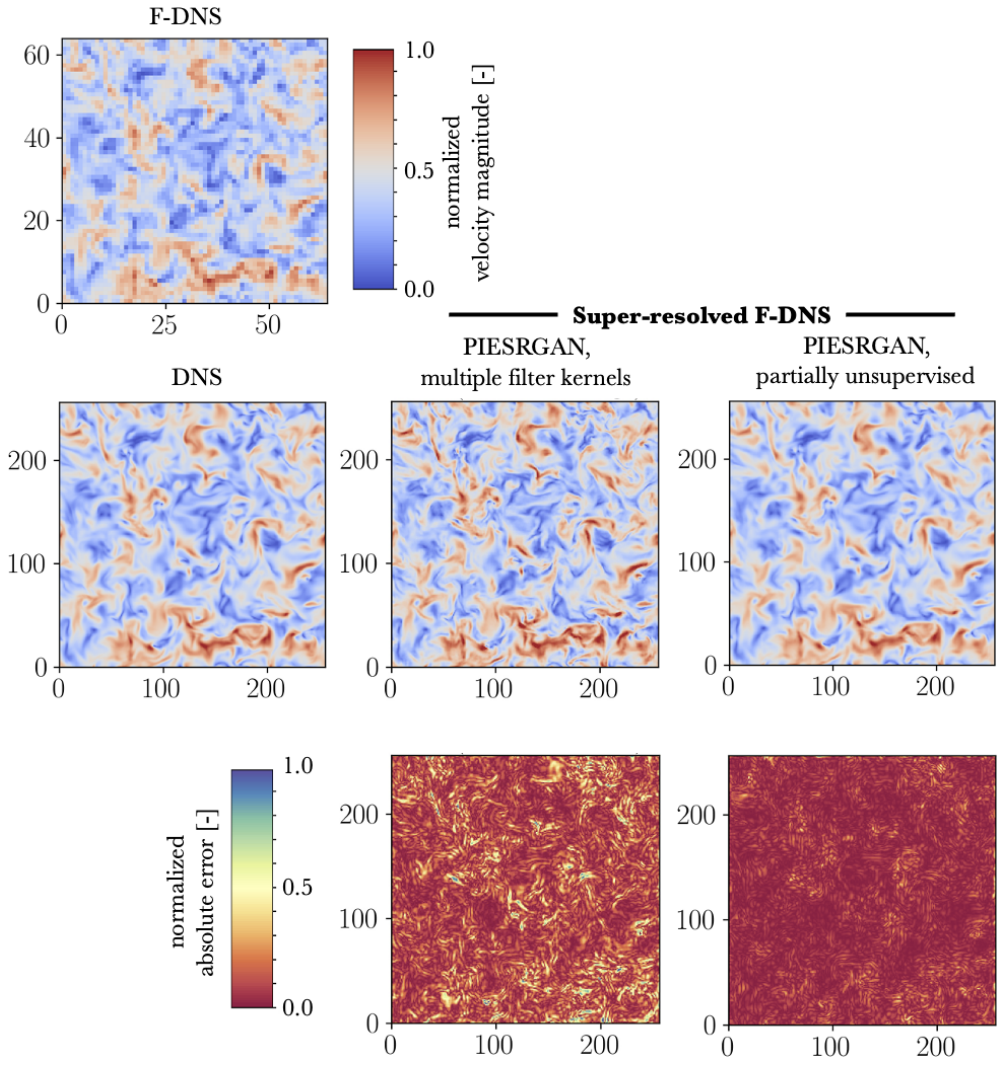

Figure 8: 2D slices of instantaneous normalized velocity magnitude and the absolute error of SR fields versus DNS. The normalized absolute error is $\hat{E} = E/\max(E_{\mathrm{DNS,SR}_1}, E_{\mathrm{DNS,SR}_2})$, where $E = \left|u_{\mathrm{DNS}} - u_{\mathrm{SR}_{1,2}}\right|$, and the subscripts 1 and 2 indicate the multiple-filter-kernel and semi-supervised training approaches, respectively.

normalized velocity gradient PDF when the partially unsupervised training is employed. The significant improvements at both large and small scales indicate that the unsupervised training step, possible only with GAN-based models, helps to generalize the model to diverse input fields.

The role of the discriminator can be further shown by disabling either the adversarial or continuity loss in equation 12. With this, the generator is trained in a fully unsupervised manner, relying solely on feedback from the continuity equation (1st option) or solely on feedback from the discriminator network (2nd option). Figure 10 compares the possible training options showing joint PDFs of the Gaussian-filtered DNS τ_{12}^{SFS} using only $\mathrm{L}_{\mathrm{continuity}}$ loss (left), only $\mathrm{L}_{\mathrm{adversarial}}$ loss (center), and both $\mathrm{L}_{\mathrm{continuity}}$ and $\mathrm{L}_{\mathrm{adversarial}}$ losses (right) during the unsupervised training phase. The PIESRGAN-reconstructed field using only the $\mathrm{L}_{\mathrm{adversarial}}$ loss during unsupervised training is statistically similar that obtained when both losses are employed. Conversely, the reconstructed field obtained using only the $\mathrm{L}_{\mathrm{continuity}}$ depicts a bulkier distribution around the diagonal and a predominant shift, resulting in a consistent overestimation of the magnitude of the Reynolds stress. While the reconstruction improvement using only $\mathrm{L}_{\mathrm{adversarial}}$ is comparable to that shown in figure 9, the training process became more unstable and ultimately resulted in GAN collapse. To address this issue, the batch size and initial learning rate were decreased (the batch size was divided by half and the initial learning rate decreased by an order of magnitude) to improve the training stability [29]. Thus, the continuity loss serves to stabilize and accelerate the training process, but further investigations are needed.

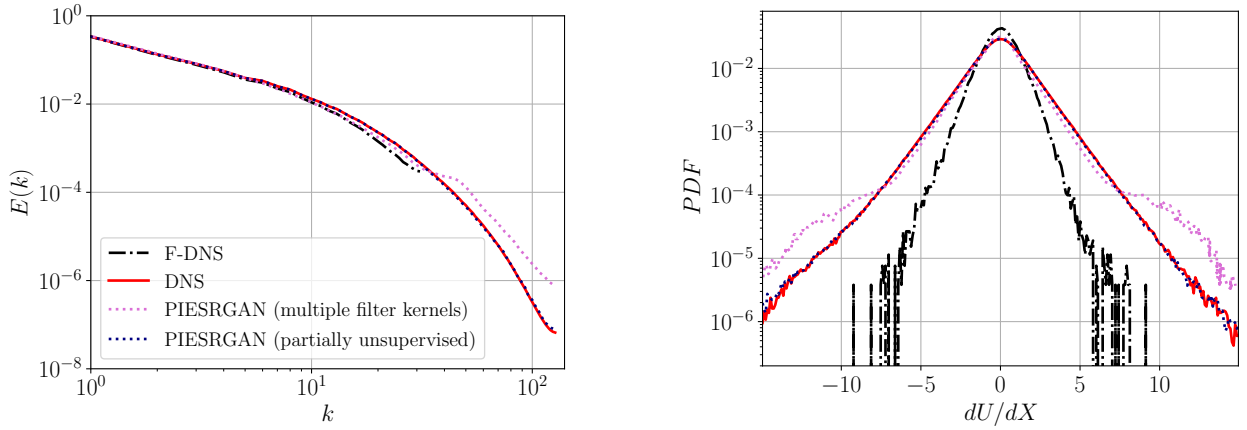

Figure 9: TKE spectra (left) and normalized velocity gradient PDFs (right) for PIESRGAN models trained for multiple filter kernels without and with unsupervised training (LES fields). Testing results are shown for Gaussian-filtered fields.

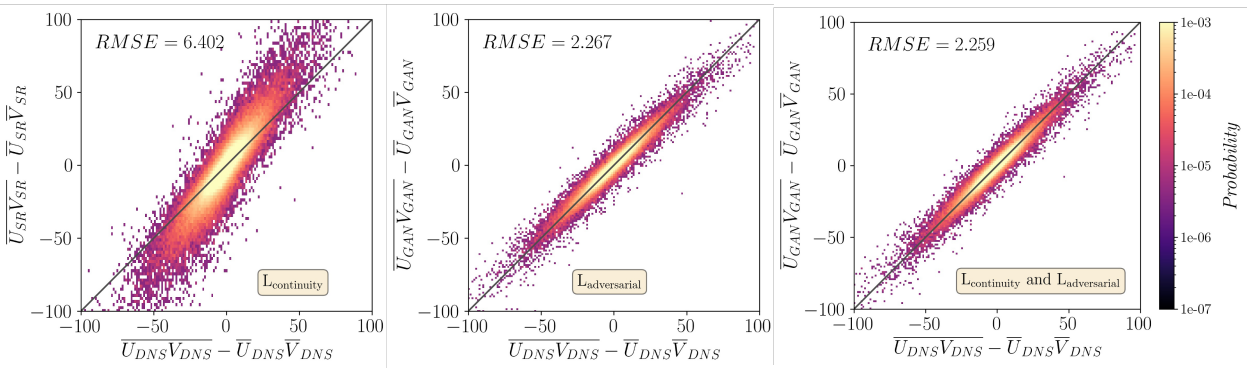

Figure 10: Joint PDFs of the Gaussian-filtered DNS τ_{12}^{SFS} using only $\text{L}_{\text{continuity}}$ loss (left), only $\text{L}_{\text{adversarial}}$ loss (center), and both $\text{L}_{\text{continuity}}$ and $\text{L}_{\text{adversarial}}$ losses (right) during the unsupervised training phase.

5.2 Different filter sizes for training and testing

We now explore the ability of SR models to reconstruct fields for different filter sizes than from those used in training. SR training typically uses fixed upsampling factors, which can result in blurry fields and/or artifacts for upsampling factors not included in the training data. This is especially critical in LES closure modeling, for different flow conditions and/or geometries can necessitate different *local* filter ratios. For this, a fixed SR upsampling factor can cause oversampling or undersampling, leading to reduced predictive capabilities.

To investigate this, SR models trained to upsample data for an LES-to-DNS-mesh ratio $\overline{\Delta}_4 = \overline{\Delta}/\Delta_{\text{DNS}} = 4$, where $\overline{\Delta}$ is the effective LES mesh resolution and Δ_{DNS} is the DNS mesh resolution, are tested for $\times 4$ upsampling of $\overline{\Delta}_8$ input data—that is, upsampling $\overline{\Delta}_8$ data to an effective LES resolution of $\overline{\Delta}_2$. All tests use box-filtered data.

Figure 11 shows TKE spectra and normalized velocity-gradient PDFs for the previously trained TSResNet and PIESRGAN models applied to this coarser Re90 input data. The PIESRGAN accurately recovers the TKE spectrum over the upsampled range, even though it does not fully recover the low-probability tails of the velocity-gradient PDF. The TSResNet model does not fare as well, with significant deviations from the DNS spectrum throughout the upsampled range. Despite these imperfections, it is clear that the PIESRGAN can upsample across different scale ranges than those used for training, at least within the self-similar scales of statistically stationary HIT. Of course, this does not imply equal success for anisotropic flows.

Multiple upsampling branches can be incorporated into the generator network to determine the ideal upsampling ratio from the input data, as applied by Lim *et al.* [50] for image-processing tasks. We now modify the generator architecture (figure 2) for multiple upsampling branches, shown in figure 12. Three branches enabling different upsampling ratios are present.

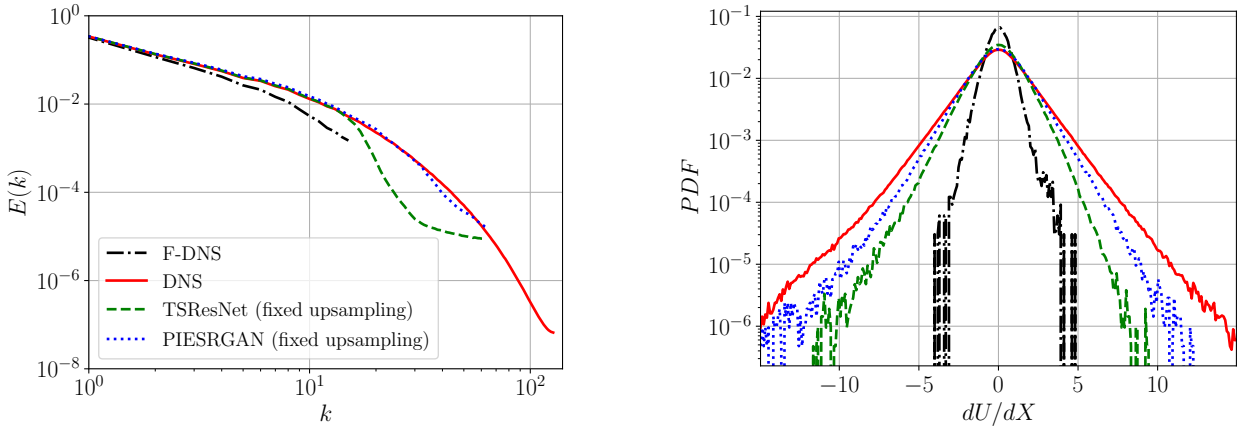

Figure 11: TKE spectra (left) and the PDF of the normalized velocity gradients (right) when both models are trained with fields filtered with a box filter with a size of four and tested on a field filtered with a box filter with a size of eight.

1. The first branch, for high-resolution inputs, contains one $\times 2$ upsampling layer.
2. The second branch, for intermediate-resolution inputs (used in Section 4), contains two upsampling layers for an overall $\times 4$ upsampling ratio.
3. The third branch, intended for low-resolution inputs, contains three upsampling layers for an overall $\times 8$ upsampling ratio.

Multiple branch employment is promising for image processing [50], though the generator needs prior information on the input size, resulting in the network only being able to handle pre-determined, known input sizes. Small upsampling factors typically result in adequate reconstruction, if incompletely resolved, with larger structures typically accurately captured. Conversely, over-upsampling can lead to significant spurious artifacts [51].

Instead, we use the discriminator to select the appropriate upsampling branch. The GAN is trained for box-filtered fields of different filter sizes ($\overline{\Delta}_2$, $\overline{\Delta}_4$, and $\overline{\Delta}_8$). During training, the correct upsampling branch is enabled based on the known filter size, and unused branches are deactivated. During testing, the generator upsamples the input field along all three branches, producing $\times 2$, $\times 4$, and $\times 8$ fields, which the trained discriminator then uses to calculate an adversarial loss (using equation 8 where $\phi_{\text{DNS}} = 0$). The upsampling factor corresponding to the lowest adversarial loss is chosen. Thus, the discriminator learns to recognize high-quality fields during training and can provide highly nonlinear feedback to the generator on the quality of the generated image.

Table 3 lists a "quality score," inversely related to the discriminator loss, for $\overline{\Delta}_2$, $\overline{\Delta}_4$, $\overline{\Delta}_8$, and $\overline{\Delta}_{16}$ input data upsampled along the $\times 2$, $\times 4$, and $\times 8$ branches. Higher values indicate higher confidence (by the discriminator) in the quality of the upsampled field. The $\overline{\Delta}_{16}$ input data is out-of-sample. The branch providing the highest quality score is selected as the most-appropriate output for arbitrary input data.

	Upsampling Factor		
Input	$\times 2$	$\times 4$	$\times 8$
$\overline{\Delta}_2$	0.93	0.46	0.08
$\overline{\Delta}_4$	0.73	0.91	0.23
$\overline{\Delta}_8$	0.53	0.69	0.84
$\overline{\Delta}_{16}$	0.17	0.34	0.59

Table 3: "Quality score" for different input filter-to-DNS-grid ratios and upsampling branches.

Figure 12 shows normalized velocity-gradient PDFs for PIESRGAN-upsampled fields using the discriminator-selected upsampling branches. It is evident that the discriminator accurately selects the appropriate branch, enabling the modified PIESRGAN to accurately recover both the large scales and the tails of the PDFs. This adaptive upsampling strategy, utilizing the discriminator, is particularly effective in scenarios where the input filter size was not present in the training data. For example, in table 3, the $\times 8$ upsampling branch is correctly selected as the most appropriate

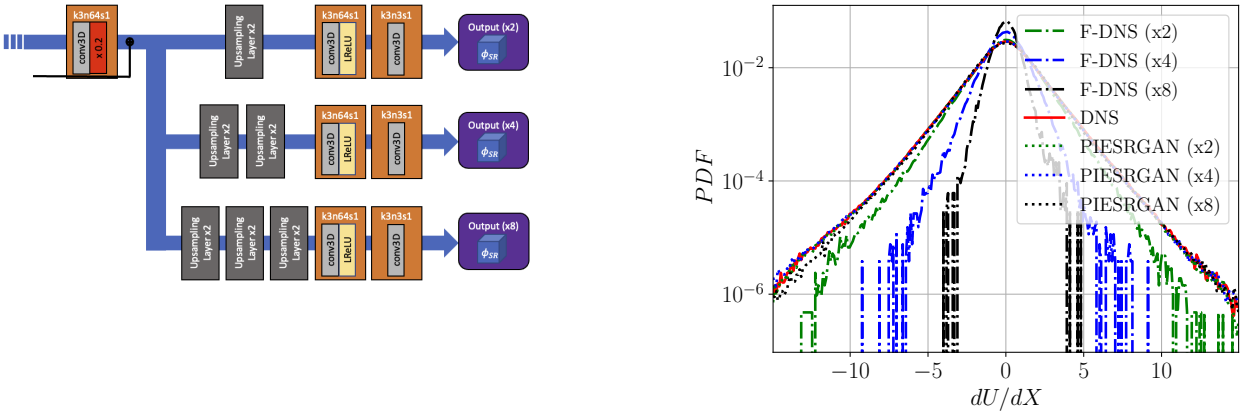

Figure 12: *Left*: Sketch of the three upsampling branches in the modified GAN generator. *Right*: PDF of the normalized velocity gradient for PIESRGAN fields reconstructed for the different upsampling factors.

available branch for the out-of-sample $\overline{\Delta}_{16}$ input data. However, it is important to note that this approach entails higher computational cost during the training compared to using a single upsampling factor (roughly 50 % higher), as more fields are employed. In conclusion, this dynamic upsampling feedback, which can only be provided in the context of a GAN-based architecture, is an effective approach to mitigating the fixed upsampling/filter size limitation.

6 Extrapolation to higher Reynolds numbers

The previous sections demonstrate that the PIESRGAN architecture can reconstruct unresolved fields more accurately than the TSResNet for in-sample flows. Moreover, the discriminator, hence adversarial training, permits greater flexibility to moderately out-of-sample input data (e.g., for different filter kernels and/or upsampling ratios), though the model must also generalize well beyond its training flows to perform accurately in *a posteriori* LES.

To test this generalization capability with equal filter kernel and size for both training and testing, thus under ideal conditions, the models trained for the lower $Re_\lambda = 90$ case (*Re90*) (section 4) are applied to forced HIT for the higher $Re_\lambda = 130$ (*Re130*) and $Re_\lambda = 350$ (*Re350*) cases (Table 1). Figure 13 compares the TKE spectra for these out-of-sample model applications to the unfiltered and box-filtered ($\overline{\Delta}_4$) DNS spectra. The $Re_\lambda = 90$-trained TSResNet

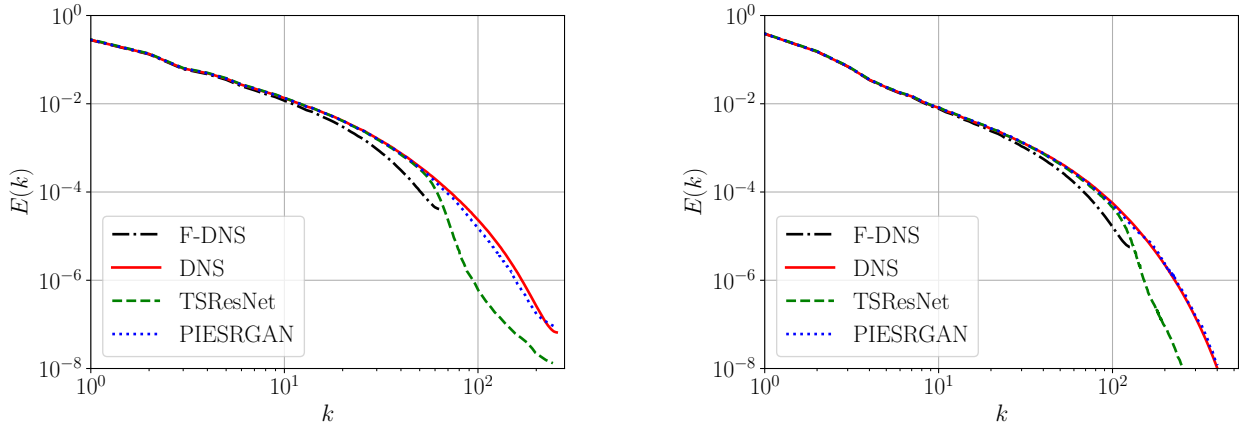

Figure 13: TKE spectra for *Re90*-trained models evaluated on higher-Reynolds-number *Re130* (left) and *Re350* (right) input data. Filtered DNS data were obtained using a $\overline{\Delta}_4$ box filter.

model fails to correctly reconstruct the subfilter scales, similar to its performance in the previous in-sample analyses, while the $Re_\lambda = 90$-trained PIESRGAN model accurately extrapolates the subfilter scales at these higher Reynolds numbers, at least outside the dissipation range. The ability of the PIESRGAN model to extrapolate the small scales of anisotropic flows remains to be studied.

The architecture can capture the scale similarity of the turbulent motions and be applied at various Reynolds numbers on similar configurations. It is worth noting that the higher-Reynolds-number reconstruction (*Re350*) is slightly more accurate than the-lower-Reynolds-number reconstruction (*Re130*). This is due to the ratio Δ/η differing slightly from the one used for training (Section 3). The jPDFs of τ_{12}^{SFS} evaluated on the PIESRGAN-reconstructed field for $Re130$ and $Re350$ inputs are shown in figure 14. In both cases, the PIESRGAN model extrapolates accurately to higher Re, with remarkable alignment of the predicted SFS stress with the DNS data. In both cases, the cross-correlation of every SFS stress tensor component computed from the fields obtained from the PIESRGAN model exceeds 90%.

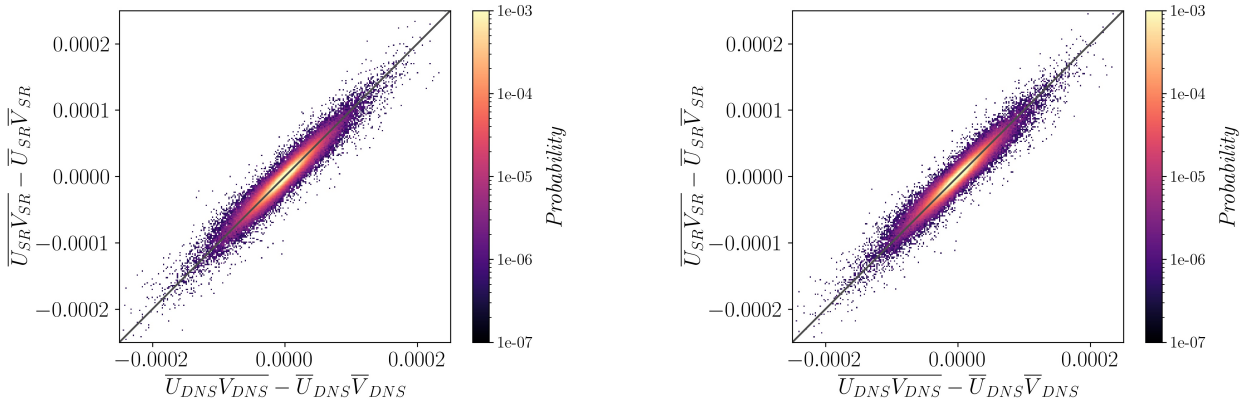

Figure 14: Joint PDFs of the box-filtered τ_{12}^{SFS} with PIESRGAN reconstruction when applied to the $Re130$ dataset (left) and the $Re350$ dataset (right).

7 Conclusion

We investigate the influence of adversarial training for super-resolution turbulence modeling. GAN-based models (PIESRGANs) are assessed against a standard, supervised CNN-based model (TSResNet). Two novel approaches are introduced that leverage the GAN model's adversarial training capabilities, enabled by its discriminator network, to enhance the generator network's accuracy and generalizability to out-of-sample inputs for different filter sizes and filter kernels. The GAN generator and TSResNet have identical model structure, hence their evaluation cost is the same despite the enhanced accuracy of the GAN model.

The two training approaches are first evaluated for in-sample data (i.e., statistically similar to the training data). In TKE spectra and velocity-gradient PDFs, the PIESRGAN architecture more accurately reconstructs the full-resolution velocity fields compared to the TSResNet model. The TSResNet model accurately captures the large-scale features, though the PIESRGAN is more consistent throughout the range of scales and velocity increments. In the context of deconvolutional modeling, the TSResNet appears to perform a soft deconvolution operation, reconstructing scales only larger than the LES filter, which would necessitate a SFS model for *a posteriori* calculations. Conversely, the PIESRGAN effectively performs both soft and hard deconvolution simultaneously. Thus a second closure term would not be needed.

Adversarial training, while computationally more expensive, is promising to enhance SR models' reconstruction of small-scale turbulence. In *a priori* SFS stress assessments, the SR models outperform the widely-used dynamic Smagorinsky model, showing significantly improved statistical alignment with the "true" filtered-DNS SFS stresses. Of the SR models, the PIESRGAN-reconstructed SFS fields are 64 % better aligned with the filtered-DNS SFS fields than those produced by the TSResNet. This is due entirely to the GAN discriminator's feedback to the generator; all other training was performed identically. The discriminator's semi-supervised training enables it to capture more complex, nonlinear relationships between the low- and high-resolution fields than a standard, supervised training approach would permit.

Data-driven deconvolution methods are known to suffer significant performance drops when applied to substantially out-of-sample data. This capability is particularly important for turbulence models to be broadly applicable. To test this, we investigated two non-consistent applications of the SR models to testing datasets: out-of-sample filter kernels and out-of-sample upscaling ratios. To assess the influence of the training filter kernel, box- and spectrally sharp-filtered SR models were applied to Gaussian-filtered DNS fields. The box/spectral-trained SR models consistently over/underestimated the SFS kinetic energy when applied to Gaussian-filtered fields and produced poorly-aligned

jPDFs of τ_{12}^{SFS}. To mitigate this behavior, both architectures were trained for a random collection of fields containing both box- and spectral-filtered data. The ensuing models have slightly greater reconstruction capability than the single-filter-trained models, though both still had a consistent gap to the Gaussian-filtered DNS. The TSResNet was again inaccurate for the subfilter scales, while the PIESRGAN added artifacts at high wavenumbers, with slight underprediction of filter-scale features.

To improve the GAN generator's ability to upsample unlabeled low-resolution fields (i.e., without accompanying high-resolution fields), as would occur for *a posteriori* LES, computed LES fields were provided as SR training inputs. The discriminator was employed as a feature extractor to provide feedback to the generator during training. By combining different filter kernels with computed LES data, the generator was trained to produce fields with correct high-frequency details emulating the DNS. This partially unsupervised approach permits the PIESRGAN model to handle diverse input fields better, and the issue of consistent small-scale overprediction is alleviated.

The ability of SR models to upsample out-of-sample filter-to-DNS mesh ratios was also explored. When a single upsampling factor is used for training, the reconstructed field may exhibit blurriness and artifacts, leading to inaccurate super-resolved fields and decreased predictive capability. This is particularly critical for LES closure modeling, where fixed upsampling ratios cannot be presumed in general, particularly on nonuniform meshes. To overcome this issue, two additional upsampling branches were added to the generator, with the discriminator determining which branch to apply locally. This enables the GAN model to manage input fields of potentially unknown resolution. By selecting the upsampling factor that minimizes this adversarial loss, the generator most faithfully reconstructs a given input field, at least within the range of upsampling ratios provided during training. From these analyses, it can be concluded that different filter kernels, unlabeled data (e.g., true LES data), as well as multiple filter-to-DNS mesh ratios must be provided during training to enable sufficient model robustness.

Finally, the ability of the two models to extrapolate to higher Reynolds numbers was tested. Using input fields from higher-Reynolds-number DNS, filtered to maintain the same $\overline{\Delta}/\eta$ ratio used for training, the PIESRGAN architecture more consistently reconstructed small-scale flow features than the TSResNet. This suggests that GAN-based models trained for lower Reynolds numbers can be successfully applied to slightly higher Reynolds numbers, though further testing for non-HIT flows (e.g., turbulent shear flows, turbulent reacting flows) is ongoing.

Adversarial training, although computationally more costly than standard supervised approaches, is overall successful in improving the predictive and extrapolative capabilities of the data-driven SR model. Leveraging the discriminator's feedback, a GAN-based model is able to generate more diverse and realistic samples during training that improve the model's performance and generalizability. We expect the robustness imparted by adversarial training to be effective in reducing *a posteriori* SFS modeling errors and improving the stability of SR-LES calculations, which will be essential in developing SR data-driven models for SFS closures.

Funding

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation program under the Center of Excellence in Combustion (CoEC) project, grant agreement no. *952181*, from the German Federal Ministry of Education and Research (BMBF), and the state of North Rhine-Westphalia for supporting this work as part of the NHR funding. The authors gratefully acknowledge the computing resources from the DEEP-EST project, which received funding from the European Union's Horizon 2020 research and innovation program under grant agreement no. *754304*.

Acknowledgments

The authors gratefully acknowledge the computing time granted by the NHR4CES Resource Allocation Board provided to them on the high-performance computer CLAIX at the NHR Center RWTH Aachen University. The computations for this research were performed using computing resources under project *rwth0658*. We thank Mr. R. Sedona and Dr. G. Cavallaro for their support in the porting of the application to DEEP-EST. The authors thank Mr. Shreyans Sakhare and Mr. Fabian Fröde for their exceptional support and contributions to this research project.

A Influence of numerical precision

"Mixed-precision" training has emerged as a powerful technique with significant implications for deep learning models. In conventional training, using 32-bit floating-point precision ("single precision," *FP32*) for storing model parameters incurs higher memory requirements. To address this limitation and accelerate model training, reduce memory

usage, and improve inference times, mixed-precision training has gained popularity. It combines *FP16* (16-bit floating-point precision) and *FP32*, for different parts of the neural network computation. Its major drawback is that reduced numerical precision can considerably reduce the precision of the resulting model, which is particularly detrimental for highly multiscale problems [41]. Lower precision can compromise a trained model's parameters and increase the probability that the magnitude of the loss gradients could be in the same order of the floating-point roundoff error.

To evaluate the influence of numerical precision, Figure 15 shows TKE spectra for in-sample, box-filtered, super-resolved fields from PIESRGAN models trained using mixed, single, and double precision (*FP64*). Mixed-precision training clearly leads to an accumulation of TKE at high wavenumbers, while both single and double precision are visually comparable to the DNS at these high wavenumbers. Because it is computationally less intensive, single precision is therefore chosen for the model-training results presented throughout the paper.

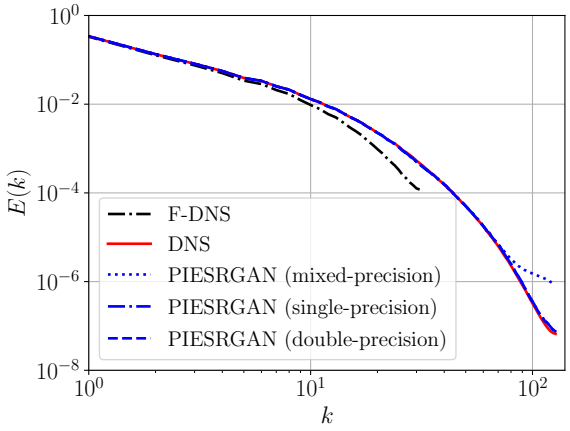

Figure 15: Comparison of the TKE spectra for in-sample box-filtered training/testing fields when different numerical-precision training are adopted.

References

[1] Masataka Gamahara and Yuji Hattori. Searching for turbulence models by artificial neural network. *Physical Review Fluids*, 2(5):054604, 2017.

[2] Paul A Durbin. Some recent developments in turbulence closure modeling. *Annual Review of Fluid Mechanics*, 50:77–103, 2018.

[3] Pierre. Sagaut and Charles. Meneveau. *Large Eddy Simulation for Incompressible Flows: An Introduction*. Number 7 in Scientific Computation. Springer, 2006.

[4] Joseph Smagorinsky. General circulation experiments with the primitive equations: I. the basic experiment. *Monthly weather review*, 91(3):99–164, 1963.

[5] Massimo Germano, Ugo Piomelli, Parviz Moin, and William H Cabot. A dynamic subgrid-scale eddy viscosity model. *Physics of Fluids A: Fluid Dynamics*, 3(7):1760–1765, 1991.

[6] Jorge Bardina. *Improved turbulence models based on large eddy simulation of homogeneous, incompressible, turbulent flows*. Stanford University, 1983.

[7] S. Stolz and N. A. Adams. An approximate deconvolution procedure for large-eddy simulation. *Physics of Fluids*, 11(7):1699–1701, 1999.

[8] Yan Zang, Robert L. Street, and Jeffrey R. Koseff. A dynamic mixed subgrid-scale model and its application to turbulent recirculating flows. *Physics of Fluids A: Fluid Dynamics*, 5(12):3186–3196, 1993.

[9] Alberto Scotti and Charles Meneveau. Fractal model for coarse-grained nonlinear partial differential equations. *Phys. Rev. Lett.*, 78:867–870, Feb 1997.

[10] Marc Terracol, Pierre Sagaut, and Claude Basdevant. A multilevel algorithm for large eddy simulation of turbulent compressible flows. *Comptes Rendus de l'Académie des Sciences - Series IIB - Mechanics-Physics-Astronomy*, 328(1):81–86, 2000.

[11] Matthias Ihme, Wai Tong Chung, and Aashwin Ananda Mishra. Combustion machine learning: Principles, progress and prospects. *Progress in Energy and Combustion Science*, 91:101010, 2022.

[12] Ricardo Vinuesa and Steven L Brunton. The potential of machine learning to enhance computational fluid dynamics. *arXiv:2110.02085*, 2021.

[13] Kai Fukami, Koji Fukagata, and Kunihiko Taira. Super-resolution analysis via machine learning: a survey for fluid flows. *Theoretical and Computational Fluid Dynamics*, 2023.

[14] Kai Fukami, Koji Fukagata, and Kunihiko Taira. Super-resolution reconstruction of turbulent flows with machine learning. *Journal of Fluid Mechanics*, 870:106–120, 2019.

[15] Kai Fukami, Koji Fukagata, and Kunihiko Taira. Assessment of supervised machine learning methods for fluid flows. *Theoretical and Computational Fluid Dynamics*, 34(4):497–519, 2020.

[16] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[17] Pranshu Pant and Amir Barati Farimani. Deep learning for efficient reconstruction of high-resolution turbulent dns data. *arXiv preprint arXiv:2010.11348*, 2020.

[18] Bo Liu, Jiupeng Tang, Haibo Huang, and Xi-Yun Lu. Deep learning methods for super-resolution reconstruction of turbulent flows. *Physics of Fluids*, 32(2):025105, 2020.

[19] Zhideng Zhou, Binglin Li, Xiaolei Yang, and Zixuan Yang. A robust super-resolution reconstruction model of turbulent flow data based on deep learning. *Computers & Fluids*, 239:105382, 2022.

[20] Malik Hassanaly, Andrew Glaws, Karen Stengel, and Ryan N King. Adversarial sampling of unknown and high-dimensional conditional distributions. *Journal of Computational Physics*, 450:110853, 2022.

[21] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[22] Zhiwen Deng, Chuangxin He, Yingzheng Liu, and Kyung Chun Kim. Super-resolution reconstruction of turbulent velocity fields using a generative adversarial network-based artificial intelligence framework. *Physics of Fluids*, 31(12):125111, 2019.

[23] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

[24] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *Proceedings of the European conference on computer vision (ECCV) workshops*, pages 0–0, 2018.

[25] Sangseung Lee and Donghyun You. Data-driven prediction of unsteady flow over a circular cylinder using deep learning. *Journal of Fluid Mechanics*, 879:217–254, 2019.

[26] Mathis Bode, Michael Gauding, Zeyu Lian, Dominik Denker, Marco Davidovic, Konstantin Kleinheinz, Jenia Jitsev, and Heinz Pitsch. Using physics-informed enhanced super-resolution generative adversarial networks for subfilter modeling in turbulent reactive flows. *Proceedings of the Combustion Institute*, 38(2):2617–2625, 2021.

[27] Mathis Bode, Michael Gauding, Dominik Goeb, Tobias Falkenstein, and Heinz Pitsch. Applying physics-informed enhanced super-resolution generative adversarial networks to turbulent premixed combustion and engine-like flame kernel direct numerical simulation data. *Proceedings of the Combustion Institute*, 2023.

[28] Ludovico Nista, Christoph K. D. Schumann, Temistocle Grenga, Antonio Attili, and Heinz Pitsch. Investigation of the generalization capability of a generative adversarial network for large eddy simulation of turbulent premixed reacting flows. *Proceedings of the Combustion Institute*, 39(4):5279–5288, 2023.

[29] Ludovico Nista, Christoph Schumann, Temistocle Grenga, Amir N. Karimi, Gandolfo Scialabba, Mathis Bode, Antonio Attili, and Heinz Pitsch. Turbulent mixing predictive model with physics-based generative adversarial network. In *10th European combustion meeting*, pages 460–465, 2021.

[30] Temistocle Grenga, Ludovico Nista, Christoph K. D. Schumann, Amir Karimi, Gandolfo Scialabba, Antonio Attili, and Heinz Pitsch. Predictive data-driven model based on generative adversarial network for premixed turbulence-combustion regimes. *Combustion Science and Technology*, 0(0):1–24, 2022.

[31] Jonathan F. MacArt, Temistocle Grenga, and Michael E. Mueller. Effects of combustion heat release on velocity and scalar statistics in turbulent premixed jet flames at low and high Karlovitz numbers. *Combust. Flame*, 191:468–485, 2018.

[32] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for GANs do actually converge?, 2018.

[33] Olivier Desjardins, Guillaume Blanquart, Guillaume Balarac, and Heinz Pitsch. High order conservative finite difference scheme for variable density low mach number turbulent flows. *Journal of Computational Physics*, 227, 2008.

[34] Thomas S Lundgren. Linearly forced isotropic turbulence. *Center for Turbulence Research Annual Research Briefs 2003*, 2003.

[35] Stephen B Pope. *Turbulent flows*. Cambridge university press, 2000.

[36] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, 2019.

[37] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[38] Alexander Sergeev and Mike Del Balso. Horovod: fast and easy distributed deep learning in tensorflow. *arXiv preprint arXiv:1802.05799*, 2018.

[39] Dami Choi, Christopher J Shallue, Zachary Nado, Jaehoon Lee, Chris J Maddison, and George E Dahl. On empirical comparisons of optimizers for deep learning. *arXiv preprint arXiv:1910.05446*, 2019.

[40] Gabriel Freytag, João VF Lima, Paolo Rech, and Philippe OA Navaux. Impact of reduced and mixed-precision on the efficiency of a multi-GPU platform on CFD applications. In *Computational Science and Its Applications–ICCSA 2022 Workshops: Malaga, Spain, July 4–7, 2022, Proceedings, Part IV*, pages 570–587. Springer, 2022.

[41] Tomas Hrycej, Bernhard Bermeitinger, and Siegfried Handschuh. Training neural networks in single vs. double precision. In *Proceedings of the 14th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management*. SCITEPRESS - Science and Technology Publications, 2022.

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

[43] Mathis Bode. Ai super-resolution subfilter modeling for multi-physics flows. In *Proceedings of the Platform for Advanced Scientific Computing Conference*, pages 1–10, 2023.

[44] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[45] Alexia Jolicoeur-Martineau. The relativistic discriminator: a key element missing from standard GAN. *arXiv preprint arXiv:1807.00734*, 2018.

[46] Antonio Attili and Fabrizio Bisetti. Fluctuations of a passive scalar in a turbulent mixing layer. *Phys. Rev. E*, 88:033013, Sep 2013.

[47] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. Learning deep CNN denoiser prior for image restoration. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3929–3938, 2017.

[48] Tom Tirer and Raja Giryes. Super-resolution via image-adapted denoising CNNs: Incorporating external and internal learning. *IEEE Signal Processing Letters*, 26(7):1080–1084, 2019.

[49] Qingyi Zhao, Guodong Jin, and Zhideng Zhou. Deep learning method for the super-resolution reconstruction of small-scale motions in large-eddy simulation. *AIP Advances*, 12(12):125304, 2022.

[50] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017.

[51] Li Xu, Jimmy SJ Ren, Ce Liu, and Jiaya Jia. Deep convolutional neural network for image deconvolution. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27, pages 1–13. Curran Associates, Inc., 2014.