Theor. Comput. Fluid Dyn.
https://doi.org/10.1007/s00162-024-00716-y

ORIGINAL ARTICLE

**Thibault Maurel-Oujia · Suhas S. Jain · Keigo Matsuda ·**
**Kai Schneider · Jacob R. West · Kazuki Maeda**

# Neural network models for preferential concentration of particles in two-dimensional turbulence

Received: 12 March 2024 / Accepted: 23 July 2024
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2024

**Abstract** Cluster and void formations are key processes in the dynamics of particle-laden turbulence. In this work, we assess the performance of various neural network models for synthesizing preferential concentration fields of particles in turbulence. A database of direct numerical simulations of homogeneous isotropic two-dimensional turbulence with one-way coupled inertial point particles, is used to train the models using vorticity as the input to predict the particle number density fields. We compare encoder–decoder, U-Net, generative adversarial network (GAN), and diffusion model approaches, and assess the statistical properties of the generated particle number density fields. We find that the GANs are superior in predicting clusters and voids, and therefore result in the best performance. Additionally, we explore a concept of "supersampling", where neural networks can be used to predict full particle data using only the information of few particles, which yields promising perspectives for reducing the computational cost of expensive DNS computations by avoiding the tracking of millions of particles. We also explore the inverse problem of synthesizing the absolute values of the vorticity fields using the particle number density distribution as the input at different Stokes numbers. Hence, our study also indicates the potential use of neural networks to predict turbulent flow statistics using experimental measurements of inertial particles.

**Keywords** Particle-laden flow · Machine learning · Turbulence · Particle clustering

T. Maurel-Oujia (✉)· Kai Schneider
Institut de Mathématiques de Marseille, Aix-Marseille Université, CNRS, Marseille, France
E-mail: tmaurelo@purdue.edu

S. S. Jain · J. R. West
Center for Turbulence Research, Stanford University, Stanford, CA, USA

S. S. Jain
Flow Physics and Computational Science Lab, George W. Woodruff School of Mechanical Engineering, Georgia Institute of Technology, Atlanta, GA, USA

K. Matsuda
Center for Earth Information Science and Technology, Japan Agency for Marine-Earth Science and Technology (JAMSTEC), Yokohama, Japan

K. Maeda
School of Aeronautics and Astronautics, Purdue University, West Lafayette, IN, USA

T. Maurel-Oujia[†]
[†] Present address: School of Aeronautics and Astronautics, Purdue University, West Lafayette, IN, USA

Published online: 09 August 2024

## 1 Introduction

The use of various tools and techniques from machine learning is becoming increasingly popular in the field of fluid mechanics (e.g., [1,2]). These tools are advancing everyday, and evaluating their potential and capabilities, especially of the deep neural networks, is critical to improve and maximize the use of data for modeling and analysis of various flows. In particular, for particle-laden flow, Siddani et al. [3] used a combination of convolutional neural networks and generative adversarial neural networks to recreate particle-resolved flow fields around a random distribution of particles, and for closure modeling for forces on particles in Siddani and Balachandar [4]. Machine learning was also used as described in Faroughi et al. [5] to predict the drag coefficient of spherical particles translating in viscoelastic fluids and in Hwang et al. [6,7], to model the forces and in Hwang et al. [8] to model collision on the nonspherical and irregular particles. For synthesizing flow data, a method using harmonic wavelet phase covariance has shown high-quality results in modeling geometric structures in turbulent flows [9] and inertial particles in turbulence [10].

These and other previous studies use machine learning to develop models for the fine-scale dynamics of the flow and particles from the macroscopic scale of information, while the motivation of the present work is to use machine learning to inform on the mesoscale clustering of inertial particles in turbulence from flow field data without the need for information on individual particles' position and velocity. Clusters of particles and void regions are characteristic features of particle-laden turbulent flows, and their prediction and understanding are essential in various industrial applications. This includes modeling radiative heat transfer in particle-laden solar receivers, impacting radar signal processing in clouds, and influencing the dispersion of pollutants in the atmosphere as well as droplets in combustion processes. For a recent review we refer to Brandt and Coletti [11].

Measuring individual particles is typically impractical. In simulations, it can be time-consuming to compute the trajectory of billions of particles. Hence, it is imperative to develop an alternative low-cost technique to predict Eulerian description of particle distributions with varying parameters, e.g., increasing the number of particles or changing the particle inertia, without the need for tracking these large number of particles. As a first step, the prediction of particle distributions with the same parameters as the given data is important. The goal of this work is to develop and test machine learning tools for this task, which is illustrated in Fig. 1. To this end, four different neural networks, encoder–decoder, U-Net, generative adversarial network (GAN), and diffusion model, are trained using direct numerical simulation (DNS) data. The aim is to synthesize one-way coupled particle number densities for different Stokes numbers from snapshots of the vorticity field of two-dimensional drift-wave turbulence, obtained by DNS [12]. The results are then compared against the point-particle DNS data and statistically assessed. In return for high-fidelity training data, which is expensive, the present results will contribute toward the development of efficient techniques that avoid expensive tracking of huge numbers of particles in DNS computations.

A complementary approach, which is also considered here, is to invert the above procedure, i.e., particle number densities are used as input for generating different flow quantities, e.g., absolute value of vorticity, as

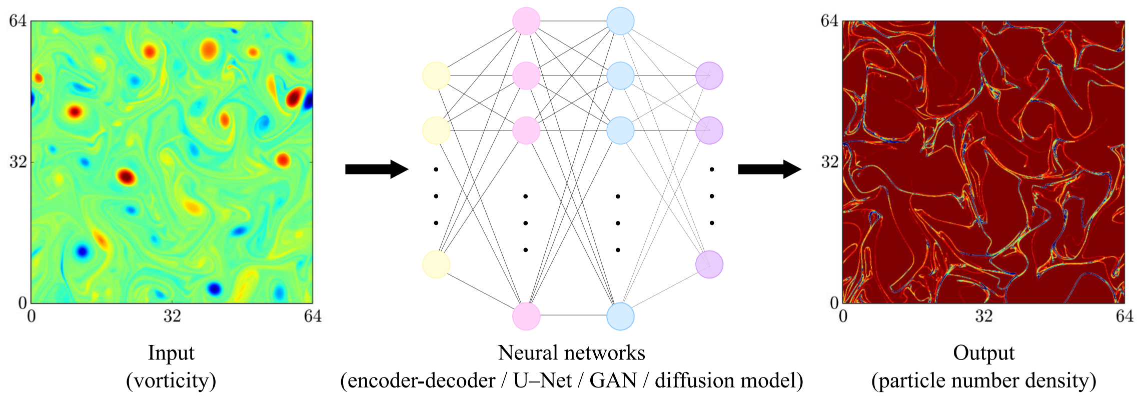
64
32
0
0
32
64
Input
(vorticity)
Neural networks
(encoder-decoder / U–Net / GAN / diffusion model)
64
32
0
0
32
64
Output
(particle number density)

**Fig. 1** A schematic illustrating the machine learning models used in this study. The left panel shows the input vorticity field, which is processed by the neural network depicted in the center, and the right panel displays the resulting particle number density field. The neural network illustration represents a conceptual overview of the model rather than its precise architecture, which is provided in Fig. 4

output that is of some interest for experimentalists. In fact, the squared vorticity yields enstrophy, which is directly related to the energy dissipation, a quantity of major importance.

The remainder of the manuscript is organized as follows. Section 2 presents the neural network methodology along with the details on the flow data set. Results are presented in Sect. 3, followed by the conclusion and perspectives for future work in Sect. 4.

## 2 Methodology

In this section, we provide details on the data set used, as well as a description of the neural network architecture and the training procedure. We assess different neural networks in this work, and the motivation behind using different networks is to assess if more complex architectures yield improved results by generating fields that have better quality with respect to the DNS data. However, a trade-off is certainly necessary in terms of computational cost and accuracy. The vorticity is used as the input to predict particle number density fields. A physical explanation behind using vorticity as the input quantity to generate the particle number density fields is the dynamics of inertial particles in turbulence. Inertial particles in turbulent flows exhibit a non-uniform spatial distribution, predominantly influenced by centrifugal forces. When the inertia of particles is small but finite, they tend to concentrate in regions of low vorticity magnitude and high strain rates, a phenomenon known as preferential concentration [13,14]. This effect arises because inertial particles in vortical regions are subjected to significant centrifugal forces, leading to their ejection. Conversely, in regions of straining with low vorticity, particles are more likely to accumulate due to reduced centrifugal expulsion. Other quantities related to inertial particle dynamics, such as divergence or curl of the particle velocity, see e.g., Maurel-Oujia et al. [15], could be likewise predicted. But in this work, we choose to focus on density, which is a more fundamental quantity.

### 2.1 Data set description

The underlying data set consists of particle position and velocity data generated by DNS of particle-laden drift-wave turbulence in the hydrodynamic regime, which is similar to 2D Navier–Stokes turbulence, as detailed in Kadoch et al. [12]. The governing equations are solved in a $L$-periodic square with a Fourier pseudo-spectral scheme, where $L = 64$. The flow reaches a statistically stationary regime, which is close to 2D homogeneous isotropic turbulence with large scale forcing and exhibits a $k^{-4}$ turbulent kinetic energy spectrum as shown in Fig. 2. The observation of a $k^{-4}$ turbulent kinetic energy spectrum aligns with findings in the literature, notably by Legras et al. [16], where such spectral slopes where reported for 2D Navier–Stokes turbulence. Uniformly distributed discrete particles are then injected into the fully developed flow and are tracked in the one–way coupled Lagrangian framework. Maxey's model [17] for inertial heavy point particles with Stokes drag is used, and the inertial dynamics is represented by the Stokes number, $St = \tau_p/\tau_f$, where $\tau_p$ is the particle relaxation time and $\tau_f$ the eddy turn over time, defined by $\tau_f = \sqrt{\nu/\epsilon}$ where $\nu$ is the kinematic viscosity and $\epsilon$ the dissipation rate. The position of the particle, denoted as $\boldsymbol{x}_p$, and its velocity, represented by $\boldsymbol{v}_p$, evolve according to the following equations:

$$d_t \boldsymbol{x}_p = \boldsymbol{v}_p, \quad d_t \boldsymbol{v}_p = -\frac{\boldsymbol{v}_p - \boldsymbol{u}_p}{\tau_p} \tag{1}$$

where $\boldsymbol{u}_p$ refers to the velocity of the fluid at the location of the particle $\boldsymbol{x}_p$.

DNS with $N_g^2 = 1024^2$ grid points is performed for a Reynolds number, $Re_\lambda = 679$. For details we refer to Kadoch et al. [12]. The number of particles $N_p$ is $10^6$ and the considered five Stokes numbers are $St = 0.05$, 0.1, 0.25, 0.5, and 1, each corresponding to a different data set. After the turbulent flow reached a statistically steady state, particles are randomly introduced into the flow, following a Poisson distribution. Particles with different Stokes numbers were tracked in an identical turbulent flow for about 13.72 eddy turn-over times. We assume that the number of particles is sufficiently large to be considered continuous, which implies that the initial positioning of particles at the flow initialization does not significantly impact their distribution.

We provide as input the vorticity fields and aim to predict the particle number density for various Stokes numbers. To facilitate faster convergence of the neural network by mitigating significant variations in output values, we perform clipping of density values exceeding 100, then divide them by 100 to ensure that the values are normalized between 0 and 1. This clipping represents 0.01% of the pixel of the input and results in

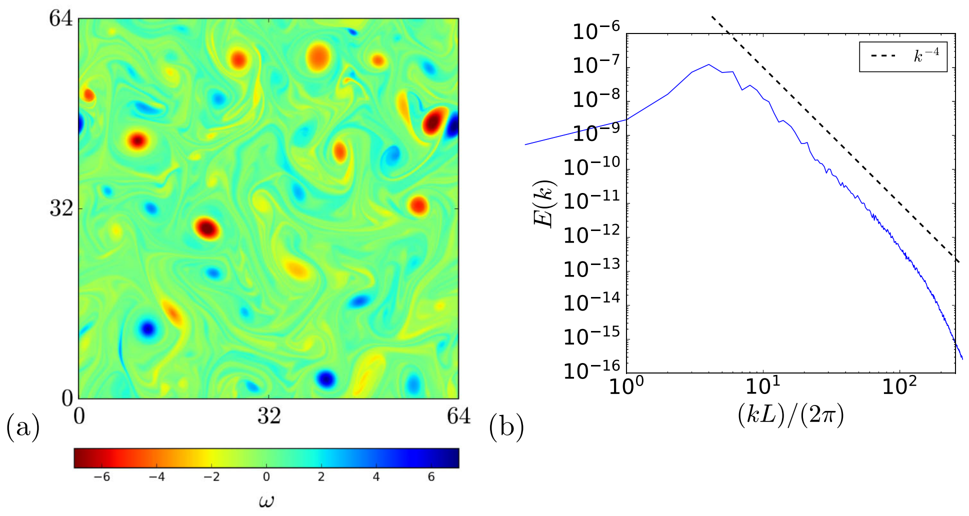

**Fig. 2** Visualization of **a** the vorticity field of DNS data and **b** the corresponding energy spectrum in the statistically stationary regime

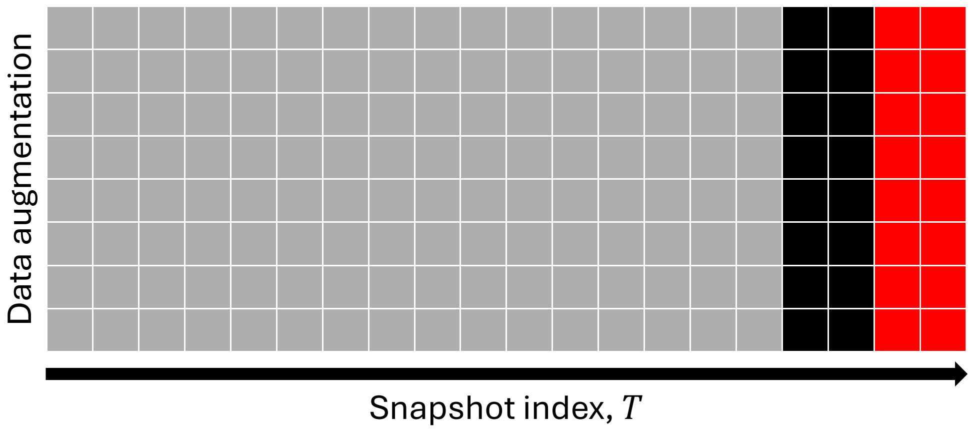

**Fig. 3** Illustration of the data set structure. The gray cells represent the training set, the black cells indicate the data that is skipped to ensure decorrelation between training and test sets, and the red cells denote the test set. The columns corresponds data augmentation operations, while the rows represent snapshots in terms of time. Adapted from Erichson et al. [18]

the removal of less than 1% of the total particle mass. The possibility of performing supersampling, that is, providing as input, not only the vorticity, but also the particle number density for an extremely low number of particles, i.e., a subsample of $10^3$ particles, is likewise explored. We are also interested in doing the inverse, i.e., predicting the absolute value of vorticity from the particle number density for different Stokes numbers.

Figure 3 shows an illustration of the data set structure. We have a total of 100 saved snapshots in order of time, and use the first 80 snapshots for the training and the last 10 for the test set. We leave a gap of 10 snapshots between the training and test sets to ensure sufficient variability and decorrelation between the two. The particle number density is calculated with a resolution of $N_g'^2 = 512^2$. Similarly, we use the vorticity field, to which average pooling has been applied, with a resolution of $512^2$ as input.

Moreover, we artificially increase the size of the training data eight times by applying different operations of transposition and rotation to the original data. This yields in total a data training set of $80 \times 8 = 640$ images and a test set of $10 \times 8 = 80$ images, for each Stokes number. All visualizations and statistics presented in the subsequent sections of this article are based on this test set.

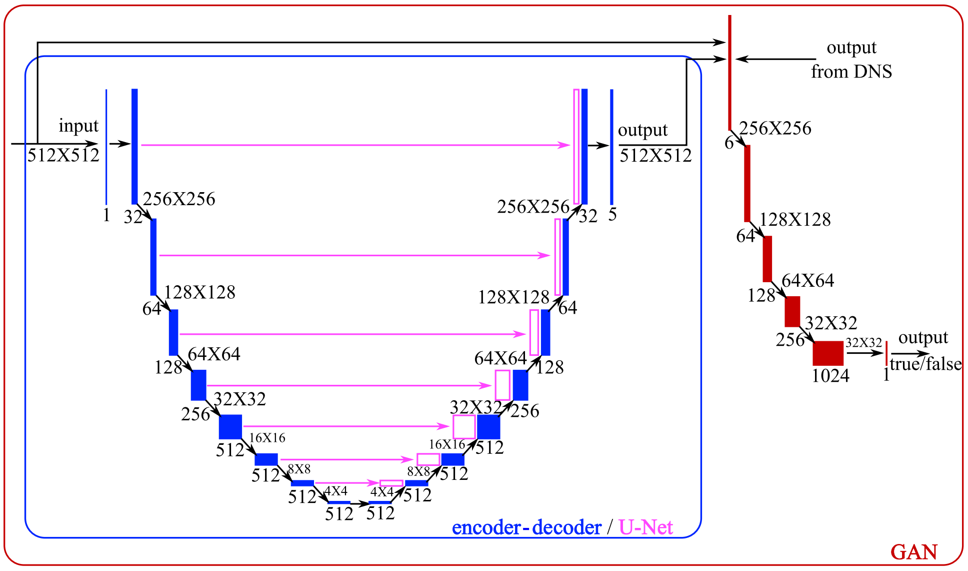

**Fig. 4** Illustration of different neural network architectures: encoder–decoder (blue), U-Net (blue and pink) and GAN (blue, pink and red). The U-Net architecture is also utilized for the diffusion model

## 2.2 Description of the neural network architecture and training

Figure 4 shows the different architectures we have tested to synthesize the number density distribution of the inertial particles for different Stokes numbers. We compare four different neural networks: encoder–decoder, U-Net, GAN, and diffusion model, each of which is a modification of the standard architecture, adapted here for the current application. An encoder–decoder [19] is an artificial neural network composed of two neural networks, an encoder and a decoder. The encoder compresses/reduces the size of the data, while the decoder reconstructs the desired output using the compressed data. The encoder–decoder is the baseline network for this study and is represented by the blocks and arrows in blue in Fig. 4. The loss function used here is the binary cross-entropy between the exact and predicted values. This is feasible because the values of the particle field are normalized between 0 and 1. A U-Net [20] is an encoder–decoder with additional skip connections between the layers of the encoder and the decoder, which have the same shape. Skip connections bypass some of the layers in the neural network to allow a connection between two distant layers, contrary to the standard connection, which are only connected to the next layer. The U-Net is represented in Fig. 4 as the blue part (encoder–decoder), to which we have added skip connections in pink. We use the same loss function (binary cross-entropy) as in the encoder–decoder. A GAN [21] is an artificial neural network composed of two neural networks, a generator and a discriminator. The generator learns to generate new data set, and the discriminator classifies if the input data is real or generated. The goal of the generator is to generate data that will be classified as real by the discriminator. We use a U-Net as the generator, as is done in Isola et al. [22]. We give as input to the discriminator the vorticity and the synthetic or exact particle number density. For the discriminator, we use the binary cross-entropy between the correct and the exact classifications (generated or real image). For the generator, we use the binary cross-entropy between the exact and the predicted values to which we add the loss of the discriminator in case we try to fool it. Thus, the objective function for the generator is given by

$$G^* = \arg\min_G \max_D \mathcal{L}_{\mathrm{GAN}}(G, D) + \mathcal{L}_{BCE}(G) \tag{2}$$

where $G$ is the generator network, $D$ is the discriminator network, $\mathcal{L}_{\mathrm{GAN}}$ is the GAN loss defined as [22]

$$\mathcal{L}_{\mathrm{GAN}}(G, D) = \mathbb{E}_{x,y}[\log(D(x, y))] + \mathbb{E}_x[\log(1 - D(x, G(x)))] \tag{3}$$

and $\mathcal{L}_{BCE}$ is the binary cross-entropy loss, defined as

$$\mathcal{L}_{BCE}(G) = -\mathbb{E}_{x,y}[y \log(G(x)) + (1 - y)\log(1 - G(x))] \tag{4}$$

with $x$ representing the input data, and $y$ the output data associated with $x$. The GAN is represented in the figure as the previous architecture to which we have added a discriminator in red. Unlike previous architectures, the diffusion model [23] operates iteratively through a series of denoising steps to incrementally refine the output. At each iteration, a noisy version of the particle number density is fed into the model, which is trained to predict the noise that has been added. The predicted noise is then subtracted from the noisy image, effectively achieving a slight denoising effect. The denoising loss is computed using an $\boldsymbol{L}^1$-norm between the denoised image and the ground truth. This process begins with a Gaussian noise and is repeated for 500 denoising steps, gradually refining the image until the final prediction is achieved with no noise. The denoising loss function for the diffusion model is defined as

$$\mathcal{L}_{\text{denoise}}(U) = \mathbb{E}_{y^t_{\text{noise}},\xi,t,x}\left[\|\xi - U(y^t_{\text{noise}}, t, x)\|_1\right] \tag{5}$$

where $t$ represents the current denoising step, $\xi$ represents the actual noise that was added to the data at step $t$, $y^t_{\text{noise}}$ is the noisy target data at step $t$, and $x$ is the conditional input data. The denoising model $U$ predicts the noise component to be subtracted. For more details on the loss function and training process, we refer to Ho et al. [23]. We employ a U-Net as the denoising model to execute these steps. The iterative process enables the model to learn complex patterns in the data, providing a more accurate and reliable prediction of the particle number density distribution for various Stokes numbers. To condition the model, we also provide the noise-free vorticity field as input.

A fundamental difference between these models lies in their predictive approaches: encoder–decoder and U-Net predict direct function mappings from the input to the output, whereas GAN and diffusion models are designed to predict transition probability distributions. The GAN achieves this through its generative process, where the generator creates data samples and the discriminator evaluates their authenticity, effectively learning the distribution of real data. Similarly, diffusion models operate by learning to reverse a gradual denoising process, thus implicitly learning the probability distribution of the original data.

In Sect. 3.1, we aim to predict particle density from vorticity. To enhance this prediction process, Sect. 3.2 introduces an additional input representing particle density for a very low number of particles ($N_p = 10^3$, $5 \times 10^2$, and $10^2$). Furthermore, Sect. 3.3 focuses on the inverse problem, i.e., predicting the absolute value of vorticity from particle number density for different Stokes numbers. For these different tasks, only the number of input and output variables changes, while the rest of the architecture is preserved, including the loss functions and other parameters.

The down-sampling convolution blocks are composed of a 2D convolution with a $3 \times 3$ filter, followed by batch normalization and a leaky rectified linear unit (LeakyReLU) activation function. To respect the spatial periodicity intrinsic to our problem, periodic padding is applied, thereby maintaining the data resolution. The sequence is repeated twice in each block, with the inclusion of a max-pooling operation at the end of the convolution block to coarsen the spatial dimensions. Conversely, the up-sampling convolutional blocks adopt a similar sequence but utilize upsampling operations to expand the spatial dimensions, effectively reversing the encoding process undertaken during down-sampling.

We use the Adam optimizer for training the networks [24]. Training was executed on an Nvidia Tesla V100 32GB GPU. The number of epochs was chosen to ensure the convergence of the different neural networks tested. For the encoder–decoder and the U-Net, we use 1000 epochs, which was more than sufficient for convergence. In the case of the GAN, due to the model's inherent instability, we selected an epoch with the lower loss value from among all the saved steps throughout the training process. For the diffusion model, we choose to train for 400 epochs, beyond which we observed signs of overfitting.

## 3 Results

In this section, we present results for different neural networks considered in this study. We first analyze the networks using visualizations in physical space and then assess them statistically, i.e., we present probability distribution functions (PDFs) and particle number density Fourier spectra and compare them with the DNS data. The density spectra are obtained after normalizing the prediction by setting the mean to zero and ensuring that the sum of all values in the spectrum equals one. This normalization prevents the mean from dominating the visualization scale of amplitude frequencies and facilitates the identification of which wavenumbers contribute to the total signal. After this normalization, the exact and predicted fields are denoted by $z$ and $\hat{z}$ respectively. The $\boldsymbol{L}^2$ error is calculated on the data in the physical space following this normalization. The $\boldsymbol{L}^2$ error is computed as the square root of the sum of the squared differences between the exact values and the predicted values, evaluated over each grid point of the Eulerian grid:

$$\boldsymbol{L}^2(z, \hat{z}) = \sqrt{\sum_{i=1}^{N'_g} \sum_{j=1}^{N'_g} (z_{i,j} - \hat{z}_{i,j})^2} \tag{6}$$

where $z_{i,j}$ and $\hat{z}_{i,j}$ are the exact and predicted values at each grid point respectively. The $\boldsymbol{L}^2$-norm of the exact values, used for comparison, is defined as follows:

$$\boldsymbol{L}^2(z) = \sqrt{\sum_{i=1}^{N'_g} \sum_{j=1}^{N'_g} (z_{i,j})^2}. \tag{7}$$

The presented results were obtained from the validation data set disjoint from the training data set, i.e., from different snapshots.

## 3.1 Prediction of preferential concentration

### *3.1.1 Some visualizations*

Figure 5 shows (a) the input vorticity field and (b) the ground truth particle number density for $St = 1$ as obtained from DNS, along with the predicted particle densities for $St = 1$ using (c) GAN, (d) diffusion model, (e) U-Net, and (f) encoder–decoder. In the vorticity field, we can observe localized zones of high rotational velocity, contrasting sharply with surrounding areas of lower vorticity. From the particle number density field, we can see that the particles are clustered in the regions of low vorticity magnitude and avoid regions of high vorticity magnitude due to the centrifugal force, similar to what is observed in Pandey et al. [25].

The prediction of the particle number density field using the GAN shows similar structures in certain regions, while also displaying some dissimilarities. Overall, a lower density of particles is observed in regions of high vorticity, as expected. However, the model also predicts the presence of particles in some areas where none were observed in the DNS field. The predicted particle number density fields also show the presence of particle clusters with thin filaments and sharp transitions from void regions to cluster regions. This is a sign that the GAN is able to generate fine-scale features accurately. This ability of GANs to predict fine-scale features has already been observed in Ledig et al. [26] in the context of image processing.

In the diffusion model prediction, we observe a substantial amount of background noise, indicating that the model has not completely succeeded in filtering out the intrinsic noise from the training process. However, there are instances where the filamentary structures are accurately positioned, akin to the results obtained with the GAN. In other cases, the alignment is less precise. Despite this, fine filaments are observed which indicates the diffusion model's ability to generate rapid density variations, which are essential for the formation of void or cluster regions.

For the U-Net prediction, the void regions are generally located where they should be, specifically in areas of high vorticity. However, we do not observe the presence of fine filaments, as seen in the predictions from the GAN and the diffusion model. In the encoder–decoder prediction, we notice a grid-like artifact; the particle number density appears diffused and blurry. The regions devoid of particles are indistinct from particle clusters. We conjecture that this could be due to the loss of information at various scales during the data compression stage.

### *3.1.2 Comparison of statistics*

Figure 6 shows the PDFs of the (a, c) particle number density and (b, d) density spectra of exact (DNS data) and predicted fields using four different architectures for (a, b) $St = 1$ and (c, d) $St = 0.05$ using different network architectures averaged on 80 test data. Results for other Stokes numbers, while not discussed in detail, can be found in the Appendix. The density spectrum is computed by taking the modulus square of the Fourier transformed density and then summed over concentric circles in wavenumber space.

For the particle number density prediction at $St = 1$, the PDF of the GAN prediction is almost perfectly superimposed with the exact DNS values. The U-Net model demonstrates a slight over-prediction of small

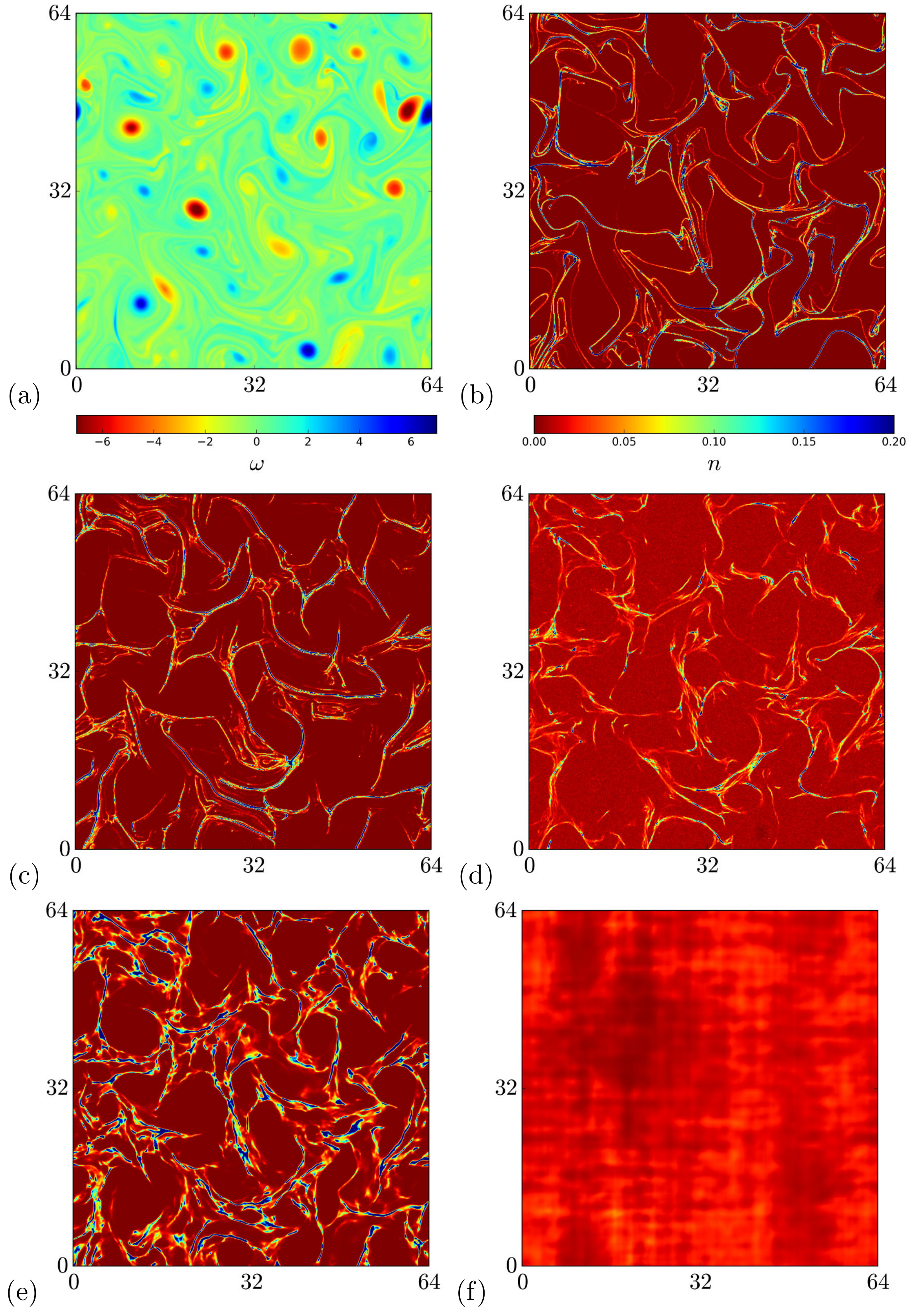

**Fig. 5** **a** Input vorticity and **b** ground truth particle number density for $St = 1$ corresponding to DNS data. Predicted particle number density using **c** GAN, **d** diffusion model, **e** U-Net and **f** encoder–decoder

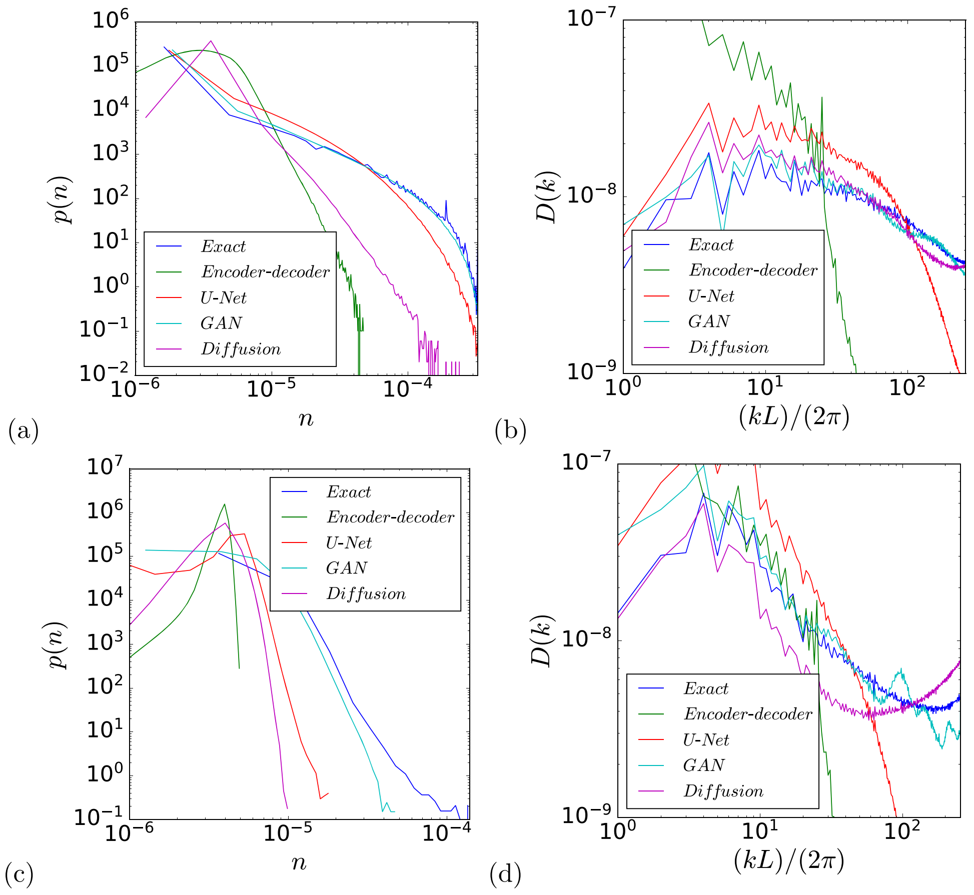

**Fig. 6** PDFs of the **a**, **c** particle number density and **b**, **d** density spectra $D(k)$ of exact (DNS data) and predicted fields using four different architectures (encoder–decoder, U-Net, GAN and diffusion model) for **a**, **b** $St = 1$ and **c**, **d** $St = 0.05$

densities, consequently under-predicting higher densities. Due to the background noise seen in Fig. 5, the diffusion model distinctly underestimates completely void regions and overestimates regions with a low but non-zero particle number. Additionally, the PDF narrows significantly for medium and high particle densities. For the encoder–decoder, the PDF declines even more rapidly, underestimating densities. Similar trends can be observed for $St = 0.05$.

Regarding the density spectra for GAN predictions at $St = 1$, the spectrum closely approximates the exact DNS values with some oscillations. A similar trend is observed for the diffusion model, with the exception that there is a slight under-prediction of high-frequency amplitudes. For the U-Net, the slope of the curve for wavenumbers less than approximately 60 is similar to those of the exact values, indicating accurate large-scale structure prediction, consistent with Fig. 5. However, past wavenumber 60, there is a rapid decrease in the amplitude. Lastly, for the encoder–decoder, the spectrum deviates significantly from the exact values, with an exceedingly rapid decay at larger wavenumbers. For $St = 0.05$, we observe behaviors similar to those described for $St = 1$, with a few exceptions. Firstly, the GAN predictions exhibit an even greater degree of oscillations. Secondly, the density spectrum for the diffusion model prediction starts to rise at higher frequencies. This unexpected rise in the spectrum is attributable to the residual noise that was not adequately eliminated during the training phase of the diffusion model. As a result, the noise becomes more predominant, skewing the signal-to-noise ratio in favor of noise and leading to this observed uptick in the density spectrum.

A portion of the error in the density spectra is due to the fact that the convolution filter is of size $N_f^2 = 3 \times 3$, which means that the maximum wavenumber that can be resolved is $(kL)/(2\pi) = N_g'/N_f = 512/3 \approx 170.67$, and the minimum length scale that can be resolved by the convolution filter is $L_{min} = LN_f/N_g'$ [27]. However, as deviations from the exact value can be observed at lower wavenumbers, we believe that this is not the dominant phenomenon affecting the accuracy of our predictions. In Matsuda et al. [28], it was observed that, unlike the flow, inertial particle clustering does not exhibit a smallest scale. This implies that no convolution filter is sufficiently small to sufficiently resolve the clustering of inertial particles.

Theoretically, in the context of one-way coupled point-particle simulations, the particle distribution is deterministic for a given simulation, provided the number of particles is sufficiently large to consider the medium as continuous. However, the particle positions are intricately linked to the fluid's historical behavior. This introduces a bias in the particle distribution; it deviates from being uniquely dependent on regions of low vorticity and vice versa. Specifically, some particles may be entirely absent from certain low-vorticity regions due to the historical flow trajectories that have previously expelled them. Consequently, predicting the particle distribution becomes a complex problem, influenced by both current conditions and history effects in the flow.

Table 1 presents a comparative analysis of $\boldsymbol{L}^2$ errors for various models and different Stokes numbers. It is observed that the GAN model consistently exhibits the lowest error across all Stokes numbers, indicating its superior ability to approximate the ground truth more accurately than the other models. The error associated with the U-Net model, although higher than that of the GAN, remains relatively close to it. In contrast, the encoder–decoder and diffusion models exhibit significantly higher errors. The error in the encoder–decoder model is mainly due to its inability to capture structures similar to the target data, as seen in Fig. 5. Similarly, the error associated with the diffusion model is also high, probably due to the background noise observed in Fig. 5, despite its ability to replicate filament structures accurately. For all models, it can be also observed that this error increases as the Stokes number decreases. Furthermore, it can be noted that for all models, the $\boldsymbol{L}^2$ errors are greater than the $\boldsymbol{L}^2$-norm. This can be attributed to the fact that while certain structures can be accurately predicted, such as with the GAN, they are not perfectly localized to the correct positions. Given the fineness of the filaments and the sparse distribution of particles, this discrepancy can contribute to an elevated error.

## 3.2 Supersampling

As previously noted, the complexity of predicting particle distribution arises not only from current flow conditions but also from the time history of the carrier flow. One approach to potentially address the computational complexity arising from a large number of particles is to explore the concept of supersampling. In this scenario, we consider a simulation with $10^3$ particles and aim to predict how the particle distribution would look like if there were $10^6$ particles. The primary advantage of such an approach could be the significant reduction in computational runtime by running simulations with a fewer number of particles. Additionally, we compare different density fields for different numbers of sub-sampled particles: $N_p = 10^3, 5 \times 10^2$, and $10^2$. Note that this approach might be limited to one-way coupled scenarios where the back-ground flow field is common regardless of the number of particles. A similar approach for reconstructing spatially high-resolution fluid flow data from its low-resolution counterpart is known as super-resolution (see e.g., [29,30]). In contrast, supersampling aims to reconstruct the complete probability distribution of particle number density from a reduced sample. This involves estimating an accurate distribution as if it were derived from a significantly larger sample size. Although both methods aim to improve data quality, they address different aspects: super-resolution enhances spatial resolution, while supersampling focuses on obtaining a precise probability distribution from a smaller dataset. Thus, supersampling could offer a promising compromise between computational cost and predictive accuracy, especially for high-resolution fluid-particle interaction simulations.

The computational cost for $10^3$ particles is 2 CPU hours, whereas for $10^6$ particles, it is 111 CPU hours, using 100 CPUs (Intel Xeon Gold 6142 at 2.6 GHz) in parallel computation. The growth is not proportional to the number of particles due to the fact that for $N_p = 10^3$, the number is too small to effectively utilize

**Table 1** Comparison of $\boldsymbol{L}^2$ errors between the DNS density field and predicted density field averaged over 80 snapshots for different models and various Stokes numbers, alongside the $\boldsymbol{L}^2$ norm of the original DNS field

| $St$ | Encoder–decoder | U-Net | GAN | Diffusion | DNS $\boldsymbol{L}^2$-Norm |
|---|---|---|---|---|---|
| 1 | $2.28 \times 10^{-4}$ | $2.64 \times 10^{-5}$ | $1.80 \times 10^{-5}$ | $5.89 \times 10^{-4}$ | $1.13 \times 10^{-5}$ |
| 0.5 | $1.43 \times 10^{-4}$ | $2.69 \times 10^{-5}$ | $2.10 \times 10^{-5}$ | $8.01 \times 10^{-4}$ | $1.43 \times 10^{-5}$ |
| 0.25 | $1.34 \times 10^{-4}$ | $4.04 \times 10^{-5}$ | $3.35 \times 10^{-5}$ | $9.43 \times 10^{-4}$ | $2.25 \times 10^{-5}$ |
| 0.1 | $2.40 \times 10^{-4}$ | $6.30 \times 10^{-5}$ | $4.86 \times 10^{-5}$ | $9.86 \times 10^{-4}$ | $3.71 \times 10^{-5}$ |
| 0.05 | $5.34 \times 10^{-4}$ | $8.17 \times 10^{-5}$ | $6.48 \times 10^{-5}$ | $1.01 \times 10^{-3}$ | $4.88 \times 10^{-5}$ |

all processors because the assignment of particles to each processor is based on their positions. However, for a large number of particles or with an optimized implementation for a non-uniform distribution of particles across different processors, the variation in computation time would be proportional to the variation in the number of particles. For $10^3$ particles, each time step takes approximately 0.36 s, and for $10^6$ particles, it takes approximately 20 s. Consequently, it takes approximately 50 times less time to compute the position of the particles for $N_p = 10^3$ compared to $N_p = 10^6$. The inference cost is approximately 0.11 s for the GAN and the U-Net, and 15 s for the diffusion model. Thus, for the U-Net or GAN, a prediction could be made at each time step without a significant increase in computational time. In the case of the diffusion model, a prediction could be made at arbitrary snapshots of interest.

Figure 7 shows the input (a) vorticity and (b) sub-sampled particle number density, ground truth (c) particle number density for $St = 1$ and predictions derived from vorticity using various models: (d) GAN, (e) diffusion model, and (f) U-Net. We observe that the particle number density distribution predicted by the GAN closely matches the exact values, even though the alignment is not identical on a pixel-by-pixel basis. Similarly, the diffusion model yields a comparable particle number density distribution; however, a persistent background noise remains evident. In the case of U-Net, we can observe finer filaments as for the original problem.

Figure 8 shows PDFs of the (a) particle number density and (b) density spectra of DNS data and predicted fields using three different architectures for $St = 1$. For the GAN, the PDF continues to be nearly perfectly superimposed on the exact DNS values. A similar trend is observed for U-Net, where the PDF now is also superimposed almost with the exact data, representing an improvement in capturing both average and high-density regions. The diffusion model shows marked improvements, especially in the prediction of high particle densities, although it still falls short in accurately capturing entirely void regions, presumably due to persistent background noise. Regarding the density spectra, no substantial variations are noted for both the GAN and the diffusion model. However, for U-Net, the predicted spectrum has moved closer to the exact DNS values, indicating improved prediction accuracy across different scales.

Figure 9 shows the PDFs of the particle number density and density spectra for $St = 1$ using the U-Net, GAN, and diffusion model architectures with $10^3$, $5 \times 10^2$, and $10^2$ sub-sampled particles. Each model was independently trained for each number of particles. It can be observed that, although the curves approach the exact values more closely as the number of sub-sampled particles increases, the differences between the curves remain relatively small.

Table 2 shows the comparison of $\boldsymbol{L}^2$ errors for the supersampling using different models for $St = 1$ with varying numbers of sub-sampled particles. The GAN model demonstrates the lowest error, indicating superior performance compared to the other models. The diffusion model exhibits the highest error, while the performance of the U-Net model lies between these two case. Comparing these results with those presented in Table 1, we can observe that for the case with $10^3$ sub-sampled particles, while all models show a reduction in error, the diffusion model benefits the most, with error reduction by one order of magnitude. For the other numbers of sub-sampled particles, we also find that the error is lower than that in Table 1, except for the U-Net model, where the error is slightly higher. As expected, when the number of sub-sampled particles is reduced, the error increases. This variation is more significant between $10^3$ and $5 \times 10^2$ particles than between $5 \times 10^2$ and $10^2$ particles. Therefore, we can conclude that a minimum number of sub-sampled particles is necessary to use the supersampling approach effectively to achieve improvement.

To summarize, we find that the predictive capability of the neural networks improve significantly when the information about a few particles are available and can be used as the input for the networks along with the vorticity data, as opposed to using only the vorticity data as the input. Hence these networks can be used as a post-processing tool to extract higher-resolution particle data by performing numerical simulations with only a few particles, which will reduce the cost of the simulations. Further investigations are necessary concerning the dependence of the input number of particles on the prediction.

**Table 2** Comparison of $\boldsymbol{L}^2$ errors between the DNS density field and predicted density field using supersampling averaged over 80 snapshots for different models for $St = 1$ with varying numbers of sub-sampled particles

| $N_p$ | U-Net | GAN | Diffusion |
|---|---|---|---|
| $10^3$ | $1.68 \times 10^{-5}$ | $1.24 \times 10^{-5}$ | $3.31 \times 10^{-5}$ |
| $5 \times 10^2$ | $2.77 \times 10^{-5}$ | $1.51 \times 10^{-5}$ | $2.56 \times 10^{-4}$ |
| $10^2$ | $2.83 \times 10^{-5}$ | $1.52 \times 10^{-5}$ | $2.91 \times 10^{-4}$ |

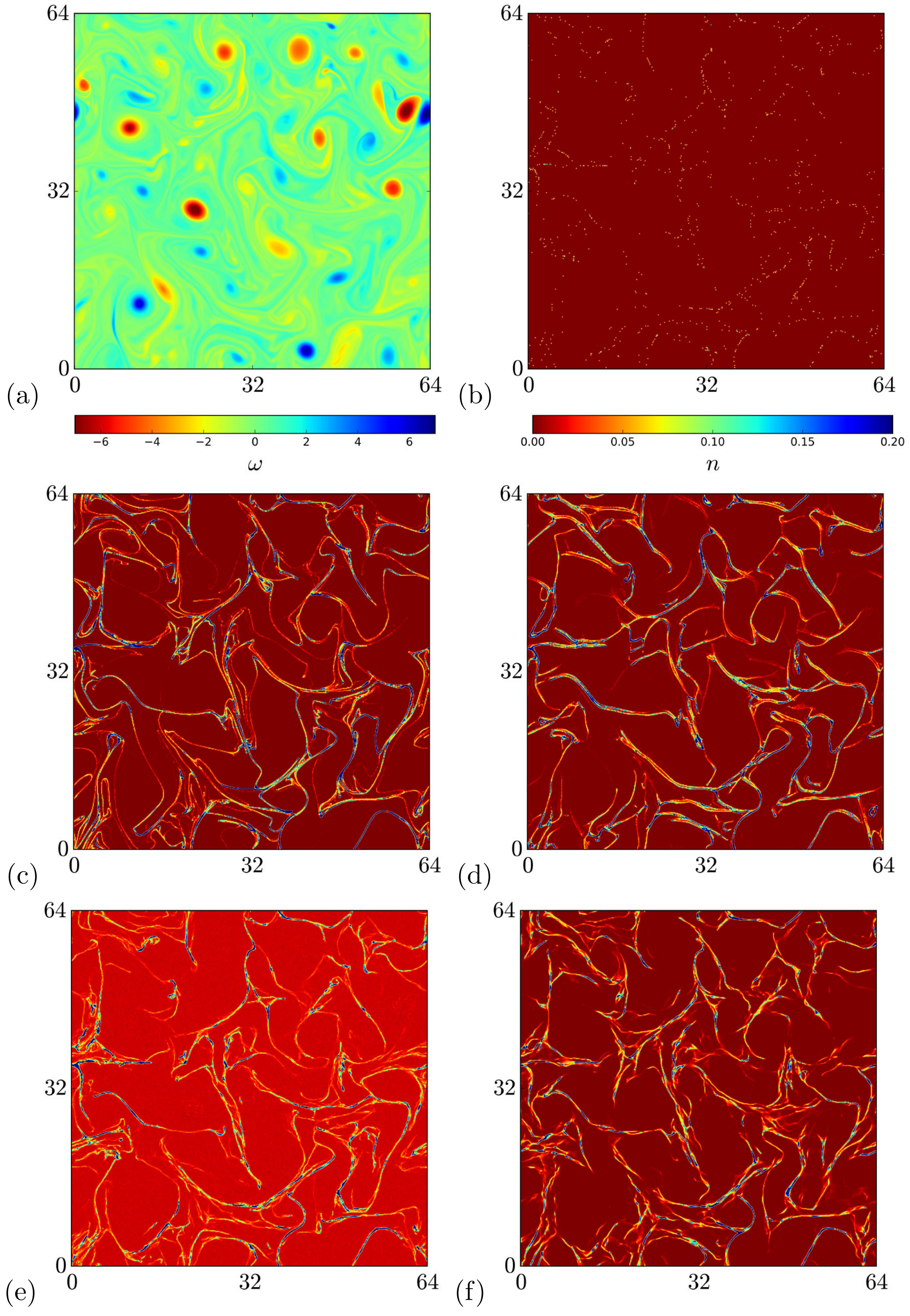

**Fig. 7** Input **a** vorticity and **b** sub-sampled particle number density, ground truth **c** particle number density for $St = 1$ and predicted from vorticity using **d** GAN, **e** diffusion model and **f** U-Net, all with supersampling

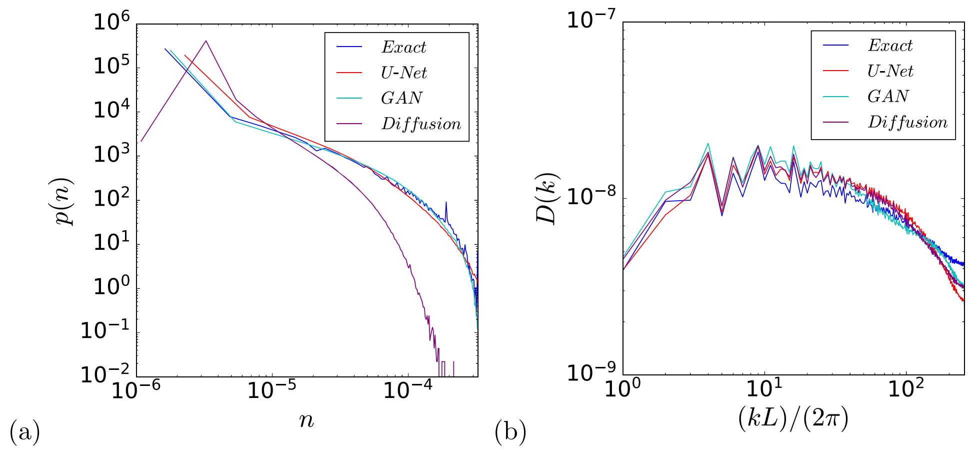

**Fig. 8** PDFs of the **a** particle number density and **b** density spectra of exact (DNS data) and predicted fields using GAN, diffusion model, and U-Net architectures for $St = 1$ with $N_p = 10^3$

## 3.3 Vorticity prediction

We are also interested in knowing the feasibility of the use of the neural networks for the enstrophy prediction from the particle number density, which could be of interest to experimentalists. To this end, we try to predict the absolute values of the vorticity field using the particle number density fields for all five Stokes numbers available to us. Note that using particle number density fields from different Stokes numbers to predict a single vorticity field, i.e. its absolute values, is possible only in the case of one-way coupled point-particle simulations. Additionally, the absolute values of the vorticity were normalized between 0 and 1 in order to use binary cross-entropy as the loss function.

Figure 10 shows the absolute values of the vorticity for (a) the exact field, and predictions from particle number density using (b) the Generative Adversarial Network (GAN), (c) the diffusion model, and (d) the U-Net architecture. We can observe that the overall structure is similar to the exact field, capturing vortex locations with accurate shape, size, and magnitude. However, in the case where two vortices are in close proximity to each other, the models tend to predict a single, larger vortex instead of the two individual structures. This behavior can be attributed to the centrifugal force that ejects particles, effectively removing data from that location and hence inhibiting the models' ability to reconstruct the absolute values of the vorticity correctly.

Figure 11 shows the enstrophy spectra of the exact vorticity field (DNS data) and the predicted absolute values of the vorticity using various network architectures. For both the GAN and the diffusion model, the spectra of the predicted data are similar to those of the exact values, with a few deviations. In contrast, the U-Net model exhibits a close approximation to the exact density spectra except at higher wavenumbers, where the predicted spectrum decreases more rapidly than it should, indicating a deviation from the exact field. We can also observe that background noise is not noticeable, unlike in the case of predicting the particle number density. Similarly to the particle number density, a portion of the error is attributable to the convolution filter size. However, unlike the particle number density, the fluid vorticity has a minimum scale, i.e., the Kolmogorov scale. To minimize the influence of the convolution filter size on the prediction, using a filter smaller than the Kolmogorov scale could be an appropriate solution.

Table 3 shows the comparative $\boldsymbol{L}^2$ errors for different models. It can be observed that the diffusion model achieves significantly lower error compared to the other models. The errors of the GAN and U-Net models are close. This shows that although the predictions of all the models are visually close, and the spectra of the predicted data are similar to those of the exact values, the diffusion model outperforms the others in predicting

**Table 3** Comparison of $\boldsymbol{L}^2$ errors between the DNS and predicted fields of absolute values of vorticity averaged over 80 snapshots for different models, alongside the $\boldsymbol{L}^2$ norm of the original DNS field

| U-Net | GAN | Diffusion | DNS $\boldsymbol{L}^2$-Norm |
|---|---|---|---|
| $8.93 \times 10^{-6}$ | $8.20 \times 10^{-6}$ | $2.00 \times 10^{-6}$ | $7.51 \times 10^{-7}$ |

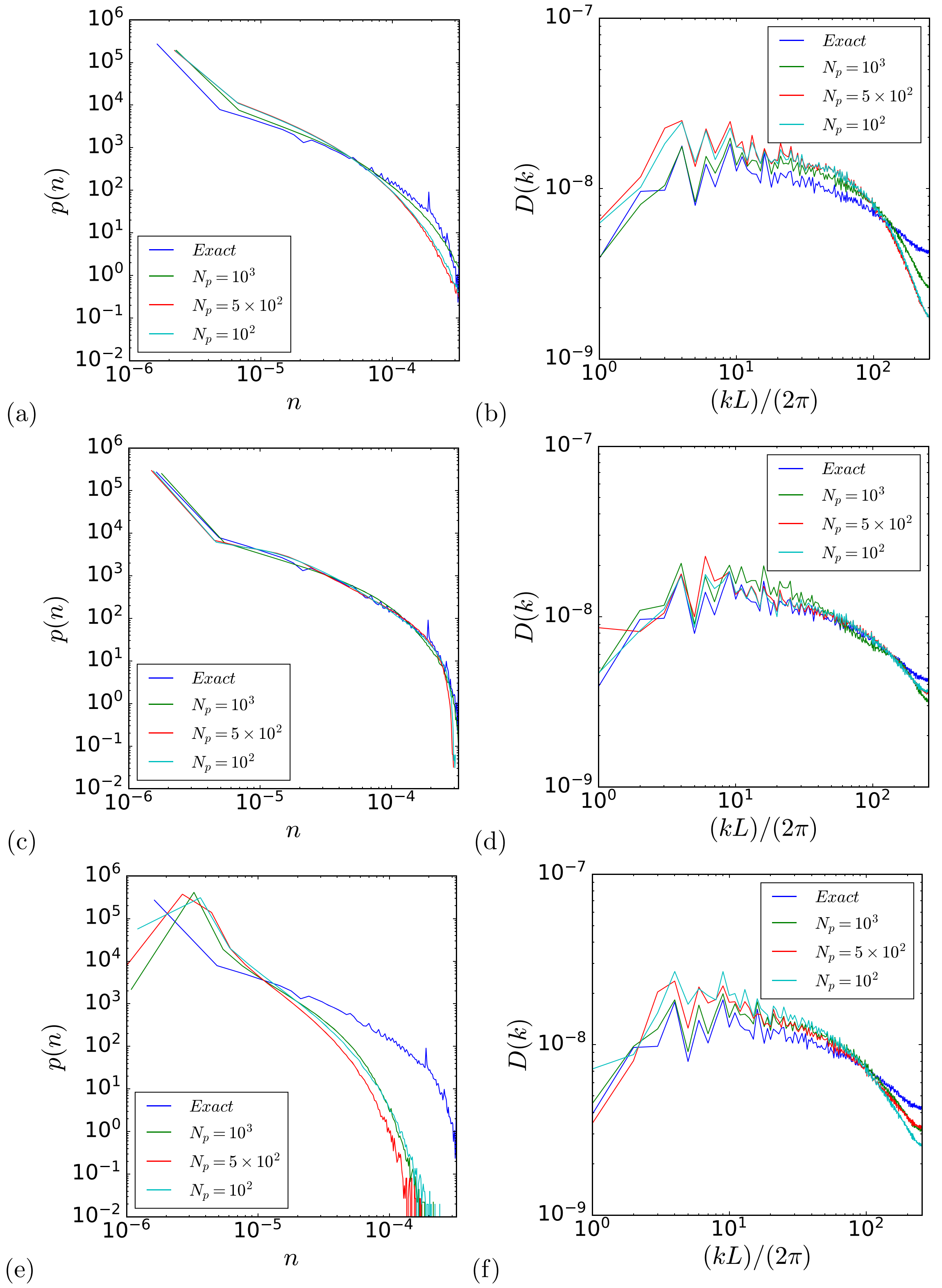

**Fig. 9** PDFs of the **a**, **c**, **e** particle number density and **b**, **d**, **f** density spectra of exact (DNS data) and predicted fields using U-Net **a**, **b**, GAN **c**, **d** and diffusion model **e**, **f** for $St = 1$ with $10^3$, $5 \times 10^2$ and $10^2$ sub-sampled particles using supersampling

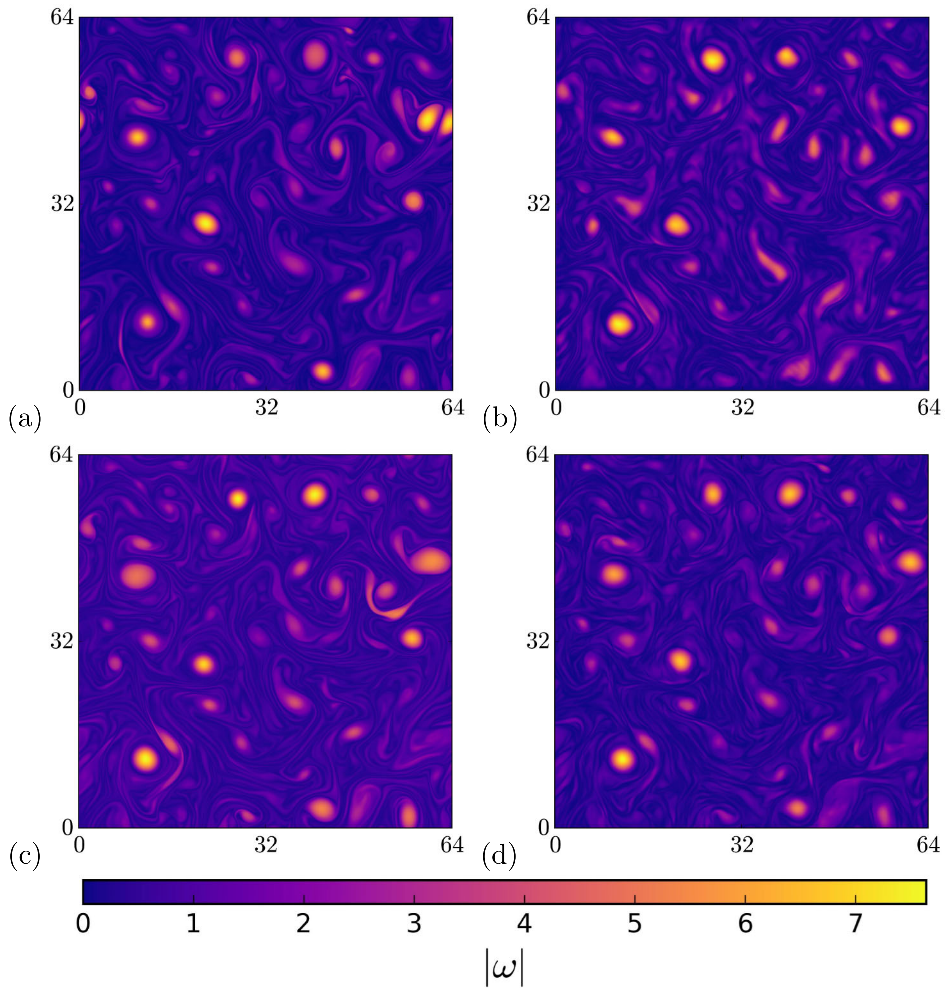

**Fig. 10** Ground truth of **a** the absolute value of the vorticity and predicted from particle number density using **b** GAN, **c** diffusion model and **d** U-Net

the absolute value of the vorticity. Furthermore, the fact that the $L^2$ errors are higher than the $L^2$-norm of the exact fields indicates that, despite the spectral similarity and visual proximity to the exact values, accuracy on a pixel-by-pixel basis is not achieved.

The present approach could be applied to particle distribution data obtained by Particle Image Velocimetry (PIV), which relies on the use of tracer particles to estimate fluid velocity. This type of machine learning method, or a derivative thereof, could be particularly useful in situations where it is possible to take photographs of a physical phenomenon and estimate the mass of the particles without being able to directly measure the fluid velocities, due to the absence of usable tracer particles. By complementing PIV, this approach would allow for the inclusion of particles of different masses in the process. Additionally, it would be advantageous in cases where adding tracer particles is impossible for various reasons, such as the distance of the objects or the fact that adding particles could perturb the experiment. This method could thus offer a valuable alternative for experiments where direct measurements of fluid velocity are not feasible with traditional PIV techniques.

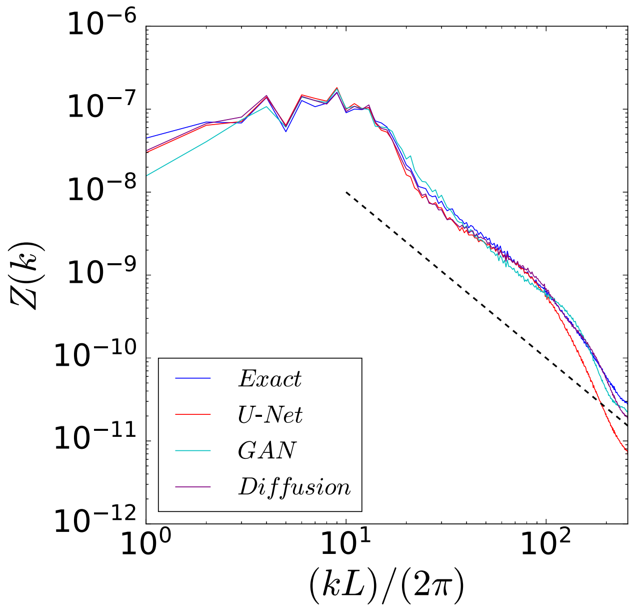

**Fig. 11** Enstrophy spectra $Z(k)$ of exact (DNS data) and predicted absolute value of the vorticity fields using different architectures. Dashed line indicates the power law $k^{-2}$

## 4 Conclusions

In this work, four neural networks for synthesizing preferential concentration fields of inertial particles in fully developed two-dimensional turbulence have been developed and assessed. Instantaneous snapshots of vorticity fields of DNS data were taken as input, and particle concentration fields were generated using encoder–decoder, U-Net, GANs, and diffusion models. The quality of prediction is quantified by comparing PDFs and density spectra of the synthesized fields with the DNS data. The best results were obtained with the GAN, showing similar cluster and void regions as present in the DNS data. This can be explained by the ability of GANs to generate fine-scale features, as well as by its generative properties. It should be noted that the results presented are valid only within the specific configuration settings of the machine learning architectures that were investigated. Furthermore, a detailed study of the parameters of the machine learning architecture would be beneficial to enhance the understanding and applicability of these findings.

Furthermore, the technique of supersampling has been explored to enhance computational efficiency. This allows machine learning models to converge more easily, providing them with information about the time history of the particles. By predicting particle distributions in simulations with fewer particles and extrapolating to higher particle counts, this approach is promising in reducing computational demands while maintaining a balance between accuracy and resource utilization. This also demonstrates the use of neural networks as a subgrid scale model, potentially in the future, in the context of large-eddy simulations where fewer particles, as opposed to a full set of particles, are used in the numerical simulations.

Inverting the input and output in the proposed procedure, we showed that reconstructing the absolute values of the vorticity field and thus enstrophy from the particle positions (e.g., obtained via PIV data) is likewise possible. This is a physically relevant task and yields a promising application for laboratory experiments. In the future, we also plan to predict fundamental quantities, such as Reynolds and Stokes numbers. Additionally, we plan to explore the extrapolation capabilities of our models by training them on a range of Stokes numbers and evaluating their performance in predicting unseen Stokes values.

**Acknowledgements** T.M.O. and K.S. acknowledge funding from the Agence Nationale de la Recherche (ANR), grant ANR-20-CE46-0010-01. K. Matsuda acknowledges financial support from JSPS KAKENHI Grant Number JP20K04298. S. S. Jain acknowledges funding from the Boeing Co. and support from the George W. Woodruff School of Mechanical Engineering at Georgia Institute of Technology. K. Maeda acknowledges funding from SRB Co., Ltd. and support from Purdue University. Benjamin Kadoch and Zetao Lin are acknowledged for providing the DNS data, Sixin Zhang for fruitful discussion and valuable advice, and Katsunori Yoshimatsu for his guidance and for facilitating a portion of the computations at the Computing Center of University of Nagoya. Centre de Calcul Intensif d'Aix-Marseille is acknowledged for granting access to its high performance computing resources.

**Funding** T.M.O. and K.S. acknowledge support from Agence Nationale de la Recherche (ANR-20-CE46-0010-01). K. Matsuda acknowledges support from JSPS KAKENHI (JP20K04298). S.S.J. acknowledges support from Boeing Co. K.Maeda acknowledges support from SRB Co., Ltd.

**Declarations**

**Author contributions** Thibault Maurel-Oujia Conceptualization, Formal Analysis, Investigation, Methodology, Software, Visualization, Writing—Original Draft, Writing—Review and Editing. Suhas S. Jain: Conceptualization, Investigation, Methodology, Visualisation, Writing—Original Draft, Writing— Review and Editing. Keigo Matsuda: Conceptualization, Funding Acquisition, Investigation, Methodology, Writing—Original Draft, Writing—Review and Editing. Kai Schneider: Conceptualization, Funding Acquisition, Investigation, Methodology, Supervision, Writing—Original Draft, Writing—Review and Editing. Jacob R. West: Conceptualization, Investigation, Writing—Original Draft. Kazuki Maeda: Conceptualization, Investigation, Methodology, Supervision, Writing—Original Draft, Writing—Review and Editing.

**Data availability** Available on request.

**Conflict of interest** The authors declare no conflict of interest related to this work.

**Ethics approval and consent to participate** Not applicable.

**Consent for publication** Not applicable.

**Materials availability** Not applicable.

**Code availability** Available on request.

## Appendix A Results for further Stokes numbers

Here, we present different results for other Stokes numbers, i.e., $St = 0.5$, 0.25 and 0.1. The results are similar and continuously approach the two extremes, i.e. $St = 1$ and 0.05, which are discussed in detail in the main text. Figure 12 shows corresponding PDFs of particle number density and density spectra for the different architectures in comparison with the DNS results.

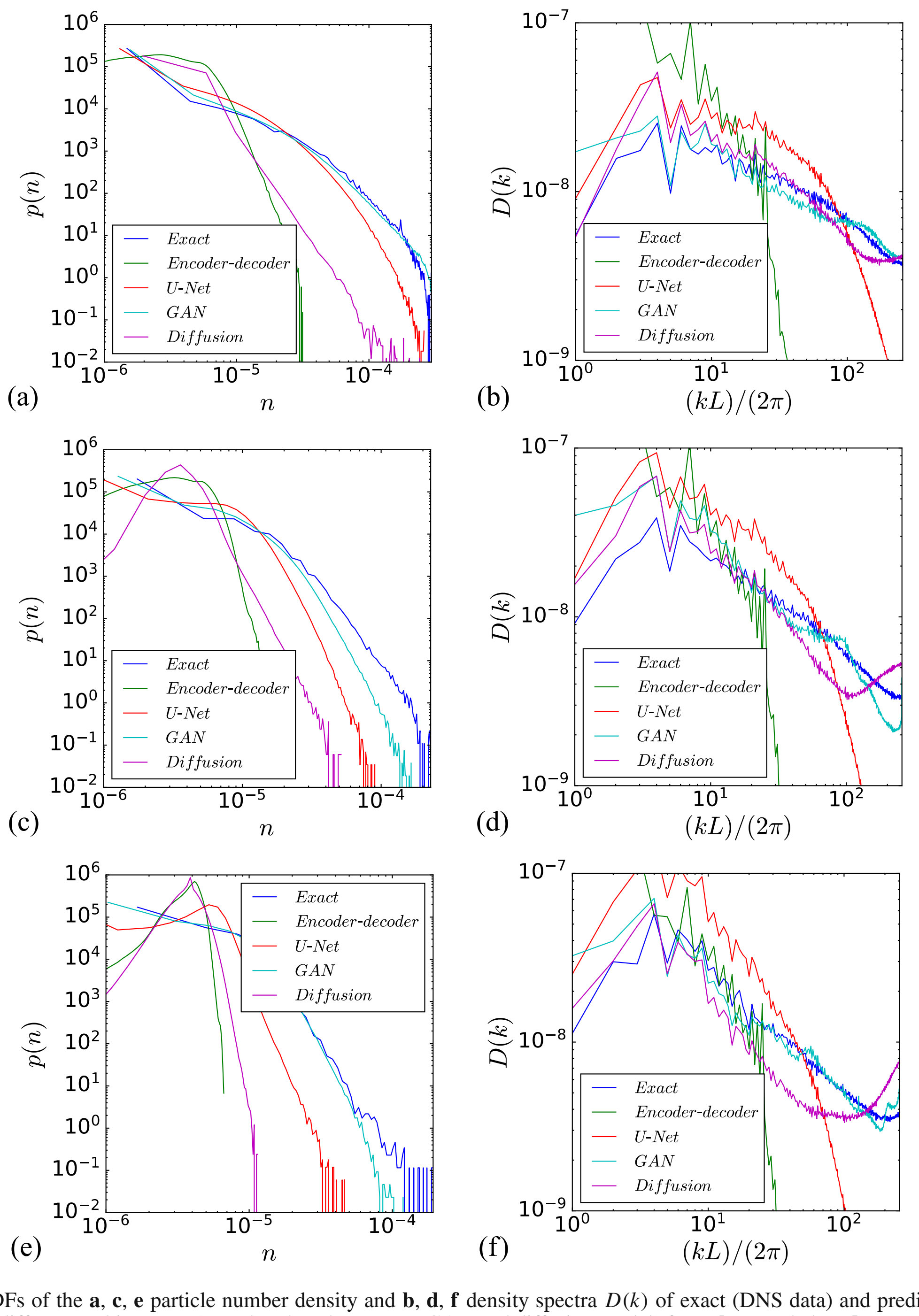

**Fig. 12** PDFs of the **a**, **c**, **e** particle number density and **b**, **d**, **f** density spectra $D(k)$ of exact (DNS data) and predicted fields using four different architectures (encoder–decoder, U-Net, GAN and diffusion model) for **a**, **b** $St = 0.5$, **c**, **d** $St = 0.25$ and **e**, **f** $St = 0.1$

## References

1. Duraisamy, K., Iaccarino, G., Xiao, H.: Turbulence modeling in the age of data. Annu. Rev. Fluid Mech. **51**, 357–377 (2019)
2. Brunton, S.L., Noack, B.R., Koumoutsakos, P.: Machine learning for fluid mechanics. Annu. Rev. Fluid Mech. **52**, 477–508 (2020)
3. Siddani, B., Balachandar, S., Moore, W.C., Yang, Y., Fang, R.: Machine learning for physics-informed generation of dispersed multiphase flow using generative adversarial networks. Theoret. Comput. Fluid Dyn. **35**(6), 807–830 (2021)

4. Siddani, B., Balachandar, S.: Point-particle drag, lift, and torque closure models using machine learning: hierarchical approach and interpretability. Phys. Rev. Fluids **8**(1), 014303 (2023)
5. Faroughi, S.A., Roriz, A.I., Fernandes, C.: A meta-model to predict the drag coefficient of a particle translating in viscoelastic fluids: a machine learning approach. Polymers **14**(3), 430 (2022)
6. Hwang, S., Pan, J., Fan, L.-S.: A machine learning-based interaction force model for non-spherical and irregular particles in low Reynolds number incompressible flows. Powder Technol. **392**, 632–638 (2021)
7. Hwang, S., Pan, J., Fan, L.-S.: Deep learning for drag force modelling in dilute, poly-dispersed particle-laden flows with irregular-shaped particles. Chem. Eng. Sci. **266**, 118299 (2023)
8. Hwang, S., Pan, J., Sunny, A.A., Fan, L.-S.: A machine learning-based particle-particle collision model for non-spherical particles with arbitrary shape. Chem. Eng. Sci. **251**, 117439 (2022)
9. Zhang, S., Mallat, S.: Maximum entropy models from phase harmonic covariances. Appl. Comput. Harmon. Anal. **53**, 199–230 (2021)
10. Brochard, A., Błaszczyszyn, B., Zhang, S., Mallat, S.: Particle gradient descent model for point process generation. Stat. Comput. **32**(3), 1–25 (2022)
11. Brandt, L., Coletti, F.: Particle-laden turbulence: progress and perspectives. Annu. Rev. Fluid Mech. **54**, 159–189 (2022)
12. Kadoch, B., del Castillo-Negrete, D., Bos, W.J., Schneider, K.: Lagrangian conditional statistics and flow topology in edge plasma turbulence. Phys. Plasmas **29**(10), 102301 (2022)
13. Squires, K.D., Eaton, J.K.: Particle response and turbulence modification in isotropic turbulence. Phys. Fluids A **2**(7), 1191–1203 (1990)
14. Squires, K.D., Eaton, J.K.: Preferential concentration of particles by turbulence. Phys. Fluids A **3**(5), 1169–1178 (1991)
15. Maurel-Oujia, T., Matsuda, K., Schneider, K.: Computing differential operators of the particle velocity in moving particle clouds using tessellations. J. Comput. Phys. **498**, 112658 (2023)
16. Legras, B., Santangelo, P., Benzi, R.: High-resolution numerical experiments for forced two-dimensional turbulence. Europhys. Lett. **5**(1), 37 (1988)
17. Maxey, M.R.: The gravitational settling of aerosol particles in homogeneous turbulence and random flow fields. J. Fluid Mech. **174**, 441–465 (1987)
18. Erichson, N.B., Mathelin, L., Yao, Z., Brunton, S.L., Mahoney, M.W., Kutz, J.N.: Shallow neural networks for fluid flow reconstruction with limited sensors. Proc. R. Soc. A **476**(2238), 20200097 (2020)
19. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. Nature **323**(6088), 533–536 (1986)
20. Ronneberger, O., Fischer, P., Brox, T.: U-net: convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-assisted Intervention, pp. 234–241. Springer, New York (2015)
21. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial networks. Commun. ACM **63**(11), 139–144 (2020)
22. Isola, P., Zhu, J.-Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1125–1134 (2017)
23. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. Adv. Neural. Inf. Process. Syst. **33**, 6840–6851 (2020)
24. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
25. Pandey, V., Perlekar, P., Mitra, D.: Clustering and energy spectra in two-dimensional dusty gas turbulence. Phys. Rev. E **100**(1), 013114 (2019)
26. Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., 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, pp. 4681–4690 (2017)
27. Zheng, Q., Li, T., Ma, B., Fu, L., Li, X.: High-fidelity reconstruction of large-area damaged turbulent fields with a physically constrained generative adversarial network. Phys. Rev. Fluids **9**(2), 024608 (2024)
28. Matsuda, K., Schneider, K., Yoshimatsu, K.: Scale-dependent statistics of inertial particle distribution in high Reynolds number turbulence. Phys. Rev. Fluids **6**(6), 064304 (2021)
29. Yasuda, Y., Onishi, R., Matsuda, K.: Super-resolution of three-dimensional temperature and velocity for building-resolving urban micrometeorology using physics-guided convolutional neural networks with image inpainting techniques. Build. Environ. **243**, 110613 (2023)
30. Fukami, K., Fukagata, K., Taira, K.: Super-resolution analysis via machine learning: a survey for fluid flows. Theoret. Comput. Fluid Dyn. **37**(4), 421–444 (2023)

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.