International Journal of ENGINE RESEARCH

Original Research Article

International J of Engine Research
2025, Vol. 26(10) 1654–1672

DOI: 10.1177/14680874251330354
journals.sagepub.com/home/jer
Sage

Machine learning benchmark for flow reconstruction in the TCC–III optical engine

Samuel Baker[1], Michael Hobley[1,2], Isabel Scherl[2], XiaoHang Fang[1,3], Felix Leach[1] and Martin Davy[1]

Abstract

We present EngineBench, the first machine learning (ML) benchmark designed for engine in-cylinder flow research. The benchmark data consist of curated particle image velocimetry (PIV) measurements previously gathered from the Transparent Combustion Chamber (TCC-III) by the General Motors University of Michigan Automotive Cooperative Research Laboratory.[1] The dataset is then leveraged in order to benchmark the performance of four ML methods for a flow reconstruction (inpainting) task. We propose large gaps at the edges of the field of view as the benchmark task in order to reflect realistic scenarios in which data are harder to obtain closer to walls, and to challenge the ability of the models to predict the turbulent flow motion with limited access to surrounding data points. Pixel-wise, vector-based and multi-scale performance metrics are used to provide broad evaluations of the models. We find that models which utilise skip connections show significantly improved performances at this task on both small and large gap sizes, due to their enhanced ability to leverage contextual information. The benchmark proposed in this paper supports the development of ML models for engine design problems, as well as PIV data enhancement more generally. In addition, the ML model comparisons allow for more informed selection of models for problems in experimental flow diagnostics. All data and code are publicly available at https://eng.ox.ac.uk/tpsrg/research/enginebench/.

Keywords

Machine learning, turbulent flow, particle image velocimetry, inpainting, flow reconstruction, benchmark, convolutional neural network

Date received: 6 December 2024; accepted: 5 March 2025

Introduction

Highly turbulent, internal flows underpin crucial technology in a range of sectors from transport and power generation to chemical processing and biofluids. Defined as flows that are bounded by ducts or channels, internal flows are needed in situations where the direction and supply of a fluid needs to be controlled,[2] with applications as diverse as the flows through the heart and lungs,[3,4] home appliances[5] and industrial furnaces.[6] In particular, in the automotive, marine and aerospace industries, these flows are used to power propulsion systems that are essential to the development of highly efficient and low-carbon transport solutions.[7,8] Experimentally, these flows are often characterized using velocity measurements from particle image velocimetry (PIV).[9] The PIV method generates velocity vectors at discrete points in the flow, such that spatially-dependent flow behaviour can be easily observed and quantified. An example of a post-processed image from PIV can be seen in Figure 1. However, conducting PIV for internal flows presents a unique set of challenges, as it is often difficult to achieve a full field of view. Gaps in the data arise due to shadowing (occlusions due to walls or other components), laser alignment issues, irregular seeding density of the tracer particles, background reflections and light scatter and strong out-of-plane motion for 2D measurements.[10,11]

[1]Department of Engineering Science, University of Oxford, Oxford, UK
[2]California Institute of Technology, Pasadena, CA, USA
[3]Schulich School of Engineering, University of Calgary, Calgary, AB, Canada

Corresponding author:
Samuel Baker, University of Oxford, Parks Road, Oxford OX1 2JD, UK.
Email: samuel.baker@eng.ox.ac.uk

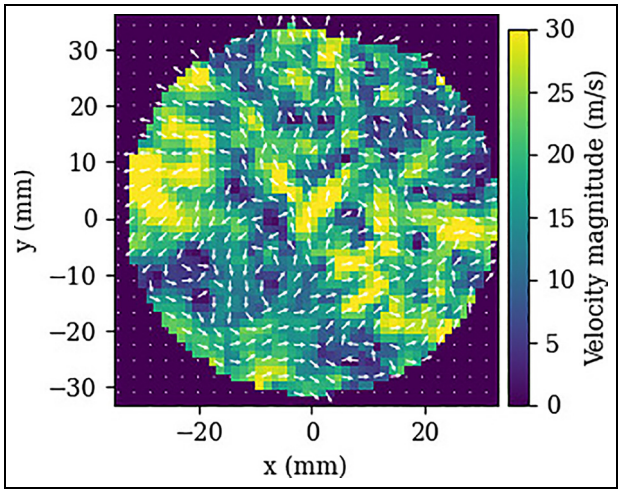

Figure 1. Example PIV image, showing a circular field of view. At each pixel the arrows show the direction of the turbulent flow and the colourmap shows the velocity magnitude.

Attempting to rectify gappy PIV data through experimental reruns may be costly or in some cases impossible, and as design work becomes increasingly digitalised, experimental data will need to be compared to or assimilated with typically clean simulation data for validation purposes[12,13] or in the construction of digital twins.[14] The pursuit of accurate digital twins and simulations of complex geometries has the potential for monumental savings in the time and costs associated with mechanical engineering design, by lessening the need to build and test prototypes.[15,16] Developing accurate methods for reconstructing full flow fields from gappy PIV data is therefore a topic of significant interest in the study of industrially-relevant flows and the development of related technologies. Furthermore, flow reconstruction models can provide valuable insight into turbulent flow behaviour more generally, by learning mappings from observable parts of the flow to obscured regions which could be leveraged towards a range of other flow reconstruction problems such as 2D–3D prediction[17] or full state reconstruction.[18,19]

The development of numerical methods that can fill gaps in spatio-temporal turbulent flow data has a history spanning several decades. Of particular note is the family of methods stemming from what came to be known as the gappy proper orthogonal decomposition (GPOD), introduced by Everson and Sirovich.[20] These methods employ the POD (akin to the principal component analysis) to identify dominant flow structures in a dataset,which are used to inform the velocity predictions inside the gaps. The predictions are updated iteratively as the number of POD modes (principal components) considered for the reconstruction is incremented. GPOD became widely-used in the field of turbulent flow diagnostics and received several updates and improvements.[21–24] Part of the reason for GPOD's popularity in the fluid mechanics community is due to the emphasis that POD methods place on dominant low-rank features, which may be analogous to the concept of coherent structures in turbulent flows.[25–27] Therefore, results from GPOD retain a degree of physical explainability. In addition, as fluid flow data are often negatively affected by noise, outliers and potentially less relevant small-scale turbulent structures,[28,29] high levels of reconstruction accuracy can be achieved by mainly focusing on these low-rank structures.[10,23]

Beyond traditional fluid mechanics research, the restoration of missing or damaged regions of an image is a well-known task in the image processing and computer vision fields, referred to therein as inpainting.[30,31] As such, data-driven methods developed in these domains can also be adapted to the problem of gappy PIV image reconstruction presented here. For example, autoencoder neural networks have been widely used in turbulent flow applications, partly because the dimensionality reduction through the latent space maintains the focus on dominant flow structures.[18,32] Convolutional neural networks (CNNs) are also widely used due to their ability to utilise the local spatial relationships inherent in turbulent flow data on grids.[33–35] In particular, the UNet architecture[36] has demonstrated success in a variety of tasks including flow field prediction and super resolution.[37–41] However, UNets are thought to exhibit some difficulty in capturing the dependency relationships of global features,[42] due to the relatively slow expansion of the receptive field through the convolutional layers.

Higher levels of performance have been reported by combining UNets with transformer modules for turbulent flow field prediction and reconstruction from limited measurements,[43–45] due to an enhanced ability to more accurately capturing the multi-scale relationships present in turbulent flows. Regarding alternative architectures, generative adversarial networks (GANs) have been used successfully in preserving multi-scale statistics of turbulent flows for super-resolution[46,47] and inpainting.[48] Physics-informed neural networks (PINNs)[49] also show promise for creating more generalizable ML models, especially for laminar and fully 3D flows.[50–52] However, the suitability of PINNs for the present 2D PIV setup is currently unclear, as the only data available are two velocity components along a 2D slice of a 3D system, which stretches the validity of the conservation equations. In addition, the spatially correlated noise introduced by the cross-correlation algorithm in the PIV process has been shown to significantly degrade the results from a PINN.[53] Other developments such as graph neural networks[54–56] and neural operators[57–59] have demonstrated success in scientific ML applications, but have not seen significant use in inpainting to date.[30,31]

In the literature on inpainting for turbulent flows, there is a noticeable lack of discussion on how artificial gaps should be created for training and testing ML models. Common methods of adding gaps to clean data include random noise,[60,61] clustered dropouts[22,23] and block gaps.[24,39,62] Rectifying random noise and clustered dropouts is often an easier task for ML models due to the large amount of spatially local

information that can be interpolated from Buzzicotti et al.[63] Conversely, due to the larger gap sizes, block gaps can be more challenging to handle. However, studies to date have only considered blocks of standard shapes and fixed orientations, which can be unrealistic and of limited use in practical scenarios where complex geometries can obscure the field of view in any number of ways. Furthermore, there are large number of inpainting models that are available to choose from, and it is not straightforwards to determine which scenarios will cause one model to perform better than another, and why. In addition, there is a lack of clarity in how different gap-handling approaches affect the results of an inpainting model. These points motivate the need for an objective benchmark to be established.

Benchmarks typically consist of an open-source dataset, and a well-defined task that can be used to objectively compare model performances, and they can be essential for developing numerical methods. For example, the ImageNet Large Scale Visual Recognition Challenge benchmark is often credited with catalysing the deep learning explosion, having facilitated the development of the famous AlexNet model.[64,65] Within the realm of turbulent flow research, several large flow physics datasets exist, including the Johns Hopkins Turbulence Database,[66] BLASTNet database[67] and the turbulence data from McConkey et al.[68] However, each of these datasets represent idealised flows over small domain sizes, which do not reflect the complex geometries and operating environments associated with physical machinery. Other databases address more practical geometries and domain sizes, such as the AirfRANS dataset for airfoil shape optimisation[69] and the Cambridge-Sandia burner for a variety of swirling stratified flows.[70] Regarding engine-specific flows, PIV datasets have been published by the Engine Combustion Network (ECN)[71] and the General Motors University of Michigan Automotive Cooperative Research Laboratory.[1] Data collected by the latter for the TCC-III combustion chamber are currently used in the benchmark established here, as the TCC-III setup was specifically designed to challenge the predictive capabilities of computational fluid mechanics (CFD) simulations, with the geometry promoting extremely complex fluidic motion via strong turbulence and high cycle-to-cycle variations.[1] In a development philosophy similar to that of CFD models, it is expected that ML models can be made more generalisable by being trained on highly challenging datasets.[72] These characteristics of the TCC data also proved valuable in the development of non-parametric dimensionality reduction approaches in the early 2010s.[73,74] A schematic of the TCC–III setup is provided in Figure 2.

This work aims to lower the barrier to entry into ML for engine researchers and makes the following contributions. A flow reconstruction target is proposed in the form of the edge gaps inpainting task, which challenges the ML models while emphasising practical relevance to engine research. A novel data augmentation method

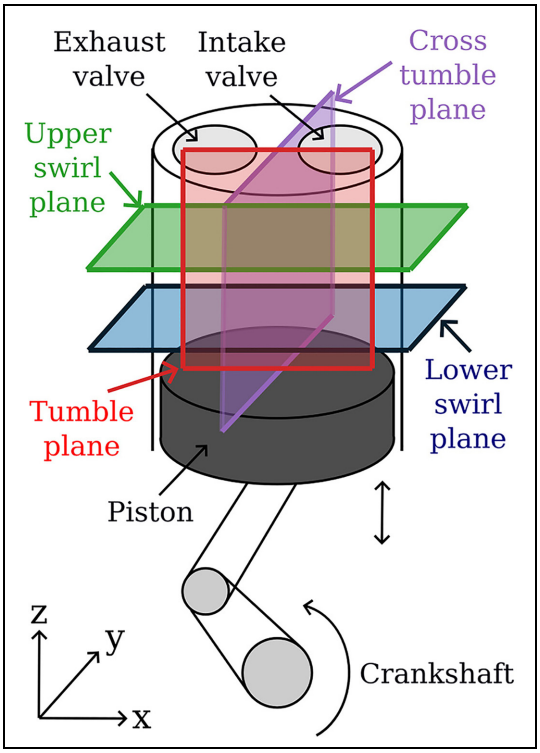

Figure 2. Schematic showing the TCC–III and associated PIV measurement planes.

designed to add random edge gaps into the training data is introduced, which outperforms the other standard methods tested here. The performances of five neural network-based models are benchmarked against the GPOD method, providing engine researchers with an objective basis for comparison that can be used to inform future model selection. The overall process followed by this work is illustrated in Figure 3. All relevant code is published along with quick-start tutorials at the web link provided in the abstract in order to promote transparency and lower the barrier to entry into ML for engine researchers. Overall, the intention of this work is to accelerate the development and adoption of inpainting models and related ML techniques within the engine research community.

Benchmark setup

Engine system

The TCC–III is a port-injected, spark-ignition, single-cylinder optical research engine with a single intake and exhaust valve each and a pancake-shaped combustion chamber consisting of a flat head and piston.[1] It has a bore × stroke of 92 × 86 mm and a geometric compression ratio of 10:1. Details of the operating conditions that produced the data used in this work are provided in Table 1. Optical access is provided via a full quartz cylinder and a 70 mm diameter flat quartz piston window. A dual-cavity Darwin Duo, Quantronix laser was used to illuminate silicone-oil seeder droplets 1 mm in diameter and images were taken with

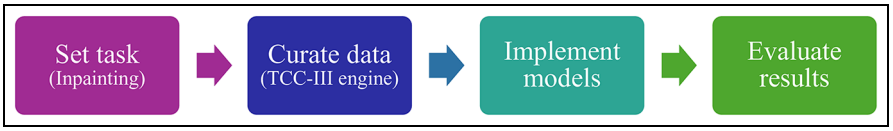

Figure 3. Flow chart depicting the benchmark creation process followed here. Descriptions of the task, data and model implementations are provided in the Benchmark setup section.

Table 1. Key EngineBench dataset information.

Parameter	EngineBench	EngineBench (LSP small)
Type	2D PIV	2D PIV
Engine	TCC–III, motored	TCCIII, motored
Pressures (kPa)	40, 95	40
Engine speeds (rpm)	800, 1300	1300
PIV planes	Lower swirl Tumble Cross-tumble	Lower swirl
Crank angles	40–705	90, 135, 180, 225, 270
# Snapshots	419,334	5205
Size	31 GB	408 MB

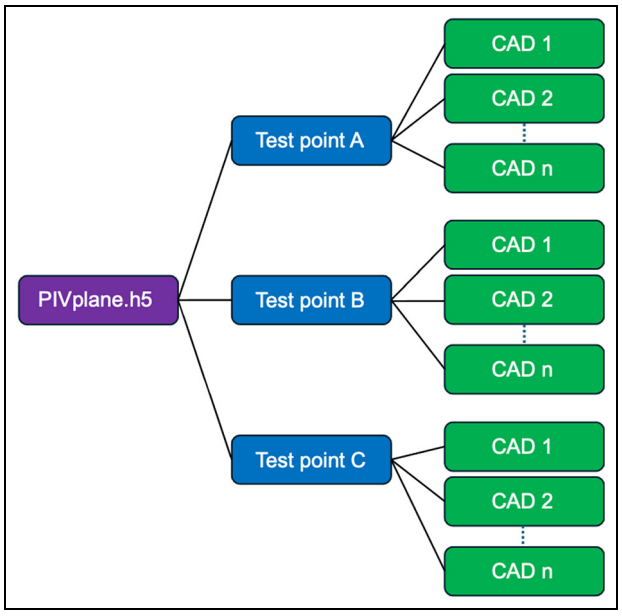

Figure 4. Generalised h5 file structure in EngineBench.

a high-speed monochrome Phantom v1610 camera (Vision Research). A multi-pass algorithm was used to process the vectors, with a decreasing interrogation window size from 128×128 to 32×32 pixels with 50% overlap. The final window size produced vectors with a spatial resolution of 1.25–1.4 mm.

The PIV data are released publicly and were created with funding by General Motors through the General Motors University of Michigan Automotive Cooperative Research Laboratory, Engine Systems Division. The TCC–III was intended to be used for providing engine-relevant data to assess and validate large-eddy simulation (LES) models, with a history dating back to the TCC-0 in the 1990s.[1] In particular, the pancake chamber design simplified meshing for CFD models and produced extremely large cyclic variations in order to challenge CFD predictive capabilities. More details on the experimental set-up are provided in Schiffman et al.[1]

Dataset

There are two databases proposed in this work, EngineBench and EngineBench LSP small. The EngineBench database consists of PIV data from motoring (i.e., unfuelled) the TCC-III engine.[1] The full database contains over 400,000 PIV images, coming to a size of 31 GB, as listed in Table 1. The dataset is stored on Kaggle as a series of h5 files, as the natively hierarchical format simplifies the chunking of data so that train/validation/test splits can be separated by specific phase angles or test points. Also, h5 files have the capability for lazy loading and the binary file format allows for efficient data storage. A diagram illustrating the hierarchical structure of each h5 file is given in Figure 4.

In order to accelerate the training times, as numerous model configurations (44 in total) were tested for the benchmark, a subset of the EngineBench data, named EngineBench LSP small, was constructed and used to generate the results. The use of a subset also makes the benchmarking results more accessible to researchers with smaller memory computers and informs practitioners on how the ML models perform with smaller datasets. The subset was constructed solely using data from the lower swirl plane (LSP), as the field of view remains constant with the changing crank angle position, simplifying the analysis. Five crank angles are extracted at phases of interest throughout the engine cycle at one operating point, as presented in Table 1. EngineBench LSP small therefore contains 5205 PIV snapshots in total and is also hosted on Kaggle, accompanied by tutorial notebooks to demonstrate how the data can be interacted with. Finally, the original spatial dimensions for each image are 50×49 pixels. Zero padding is therefore added around the edges of the images to 128×128 for compatibility with standard ML models.

Target

The goal of the benchmark was chosen to be the inpainting of so-called 'edge gaps.' In this work, edge

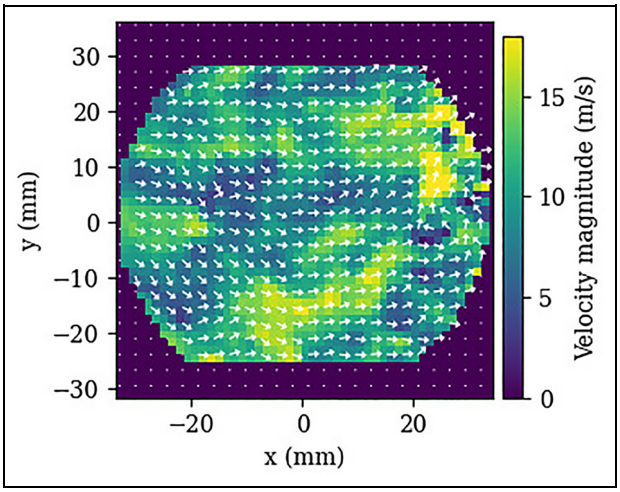

Figure 5. Example PIV image from EngineBench LSP small, with the horizontal edge gaps added at the top and bottom of the field of view.

gaps are defined as large blocks of missing data at the edges of the field of view. This type of gap was selected for a number of reasons. Firstly, they are more realistic than other types of gaps such as randomly-located blocks; edge gaps commonly occur in PIV setups that have restricted optical access due to walls.[75,76] In addition, it is especially challenging to predict the flow inside edge gaps, as there is a limited amount of local information that can inform the models. From the model's perspective, predicting the flow inside edge gaps is therefore akin to extrapolation beyond the field of view. This difficult challenge is intended to push the boundaries of what is possible with flow field data reconstruction.

A consistent test case is therefore constructed using edge gaps to benchmark the performance of the inpainting models. Two masks of a fixed shape are constructed that each remove the data at a proportion of the pixels at the edges of the field of view. A vertical mask is applied to the first half of the test set and a horizontal mask to the latter half. In addtition, two gap sizes are tested, consisting of 10% and 25% of the data missing. An example test flow field with 10% of the data missing is shown in Figure 5. Within EngineBench LSP small, all the data at 180 CAD aTDCf are held out for the test case in order to assess the generalisability of the models. The flow fields at 180 CAD are notoriously challenging to predict, as the piston is on the point of switching its direction of travel, causing the flow patterns to be highly variable.[72]

Models and training

The performance of four different model architectures is benchmarked in this study. Firstly, adaptive median filter GPOD (GPOD-MF) is chosen as a best-in-class non-parametric approach, known to outperform interpolation and other GPOD methods.[23] Secondly, the UNet model[36] is chosen due to its wide usage in turbulent flow research. Three loss functions are tested with the UNet: a mean square error (MSE) loss, a huber loss function in order to test the effect of outliers in the PIV data and a physics-based gradient loss. Further details of the loss functions are given later. Thirdly, the UNet transformer (UNETR) model[77] with an MSE loss is chosen due to the performance enhancements that have been reported due to the transformer module, with the project MONAI implementation.[78] Finally, an adapted version of the context encoder generative adversarial neural network (CE-GAN)[79] with MSE and adversarial losses is implemented due to its high performance in standard inpainting tasks[30,80] and recent usage in turbulent flows.[48,63] As the original context encoder was designed for inpainting gaps of a fixed size and location, the network architecture is modified in a similar fashion to the changes made by Li et al.[48] For the generator, an additional de-convolutional layer is included at the output to return a prediction of the same spatial dimensions as the input, forming a symmetrical autoencoder architecture. To correspond with the generator modifications, an extra convolutional layer is added at the beginning of the discriminator to handle inputs of the same size as the original data. A dropout layer with a probability of 50% is also added at the output of the discriminator, following Li et al.[48] Model summaries implemented here are provided in Table 2 for reference.

During the ML model training, the losses between the predictions and the labels are calculated across the entire image, not just inside the gap. This approach provides a number of benefits: to simplify the random gaps training process, retain the context of the broader turbulent flow and field of view and to provide practitioners with a visual representation of how the network relates the prediction inside the gap to the rest of the field, avoiding edge effects in the output. Performance metrics on the test set predictions are then reported for the central regions as well as the edge gap regions. Training hyperparameters are chosen to reflect other turbulent flow ML studies.[48,67] Training in all cases was run over 300 epochs. For the UNet and UNETR models, the learning rate was 1e–3 and multiplied by a factor of 0.5 every 50 epochs via a step scheduler. For the CE-GAN, the learning rate was 1e–4 and multiplied by 0.75 every 50 epochs. All architectural hyperparameters were retained from their original studies; however, the configurations are not guaranteed to be optimal for this specific case, as the focus of this work is on the development of an objective benchmark rather than the optimisation of the underlying ML models at this stage.

Finally, the training, validation and testing datasets were split by crank angle. As previously mentioned, 180 CAD was held out for the test set, while different permutations of the other four phase angles are then used to construct the training and validation sets, with three phases for training and one for validation. The training process for each model was run three times

Table 2. ML sizes in millions of parameters and loss functions tested.

Model	# Parameters (millions)	Loss functions
GPOD	N/A	N/A
UNet	10.5	MSE, huber and gradient
UNETR	87.3	MSE
CE-GAN	74.0	Adversarial

Table 3. Definitions of phase angle permutations that comprise the training, validation and hold-out test sets. The different permutations are denoted as A, B and C, and the corresponding phase angles are given in crank angle degrees (CAD).

Data split	A	B	C
Train	90	135	90
	135	225	225
	225	270	270
Validation	270	90	135
Test	180	180	180

with different permutations of training and validation phase angles tests, which tests the sensitivity of the model performances to the specific phases chosen for the analysis and provides the error bars for the results. The crank angle permutations are defined in Table 3. Only three permutations are considered out of the possible four as the spread across permutations was found to be acceptably low on all metrics. The resultant number of images in the train, validation and test sets are 3123, 1041 and 1041.

GPOD implementation: GPOD is performed on a single large data matrix, rather than separate training and testing matrices. Therefore, in the present study, the test data matrix at 180 CAD is stacked alongside three training data matrices according to the permutations defined in Table 3. The test edge gaps are added to the test data matrix and the gap locations are initialised using the ensemble mean from the training data matrices which do not contain any gaps. Convergence checking (CC) gaps are also implemented into the test data in a similar format to the test edge gaps; ie, the test edge gaps are extended to incorporate another 10% of the pixels in the image where the true values are known. The GPOD-reconstructed flow fields at the minimum CC L2 error are retained for analysis in this study.

Metrics

A variety of metrics are used to evaluate the model performances, in order to quantify pixelwise accuracy, vector similarity and multi-scale phenomena. The relative L2 error is used to quantify pixelwise accuracy and is calculated for true and predicted velocities u_{true} and u_{pred} as follows:

$$\text{L2} = \| u_{\text{true}} - u_{\text{pred}} \|_2 / \| u_{\text{true}} \|_2 \tag{1}$$

where u_{true} and u_{pred} are the true and model-predicted velocity vectors respectively. In addition, two vector-based metrics are used to quantify the similarity of the overall flow structures. The relevance index (RI) previously introduced is defined as:

$$\text{RI} = \frac{\langle u_{\text{true}}, u_{\text{pred}} \rangle}{\| u_{\text{true}} \|_2 \cdot \| u_{\text{pred}} \|_2} \tag{2}$$

where $\langle \cdot, \cdot \rangle$ represents the inner product. The RI varies between 1 for perfectly aligned vectors and -1 for perfectly opposite vectors. The similarity of the vector magnitudes is given by the magnitude index (MI)[81]:

$$\text{MI} = 1 - \frac{\| u_{\text{true}} - u_{\text{pred}} \|_2}{\| u_{\text{true}} \|_2 + \| u_{\text{pred}} \|_2} \tag{3}$$

with the MI varying between 1 for vectors of identical magnitude and 0 for totally disparate vector magnitudes. Finally, in order to capture the multi-scale turbulent flow features, the energy spectrum S for each image is calculated using the Fourier transform:

$$S(k) = \frac{1}{2}\left(\hat{u}\,\widetilde{(k)}\hat{u}^*(k) \right) \tag{4}$$

where $\hat{u}(k)$ is the Fourier-transformed velocity vector, $\hat{u}^*(k)$ is its complex conjugate, k is the spatial frequency wavenumber vector and $\widetilde{(\cdot)}$ represents the radial average over the vertical and horizontal frequencies.[48] The Kullback–Leibler (KL) divergence is then used to quantify the similarity between energy spectra:

$$KL\left(S_{\text{true}} \parallel S_{\text{pred}}\right) = \sum_k S_{\text{true}}(k) \left(\frac{S_{\text{true}}(k)}{S_{\text{pred}}(k)} \right) \tag{5}$$

ranging from 0 for identical distributions to infinity for a complete divergence.

Loss functions

The widely-used mean-squared-error (MSE) loss l_{mse} between two pixels at location i is given by:

$$l_{mse} = \left(u_{i,\text{true}} - u_{i,\text{pred}} \right)^2. \tag{6}$$

The Huber loss is a hybrid loss function that reduces sensitivity to outliers by applying an L1 loss to element-wise errors above a certain threshold (delta) and a quadratic loss otherwise to aid convergence. It is defined per pixel i as:

$$l_{\text{hub},\,i} = \begin{cases} 0.5\left(u_{i,\text{true}} - u_{i,\text{pred}} \right)^2 if \left| u_{i,\text{true}} - u_{i,\text{pred}} \right| < \delta \\ \delta\left(\left| u_{i,\text{true}} - u_{i,\text{pred}} \right| - 0.5\delta \right) \text{otherwise.} \end{cases} \tag{7}$$

This is then averaged over all pixels in the image pairing. A smaller value of the δ parameter increases the

Table 4. Huber loss δ tuning results for the 180 CAD test case with 10% edge gaps. One result for each setup using permutation A is reported. **Bold** typeface represents the best result.

Parameter	RI	MI	L2	KL
Central regions:				
$\delta = 5$	0.999	0.983	0.033	**0.000**
$\delta = 1$	**1.000**	0.983	0.034	**0.000**
$\delta = 0.5$	0.999	0.983	0.034	**0.000**
$\delta = 0.1$	**1.000**	**0.988**	**0.024**	**0.000**
Edge gaps:				
$\delta = 5$	0.886	0.751	0.462	0.016
$\delta = 1$	**0.896**	**0.765**	**0.445**	**0.013**
$\delta = 0.5$	0.895	0.760	0.449	**0.013**
$\delta = 0.1$	0.858	0.712	0.517	0.026

Table 5. Adversarial loss lambda tuning results for the 180 CAD test case with 10% edge gaps. One result for each setup using permutation A is reported. **Bold** typeface represents the best result.

Parameter	RI	MI	L2	KL
Central regions:				
$\lambda_{adv} = 1e{-}1$	0.709	0.578	0.709	0.099
$\lambda_{adv} = 1e{-}2$	**0.883**	**0.731**	**0.478**	**0.014**
$\lambda_{adv} = 1e{-}3$	0.709	0.616	0.809	0.082
$\lambda_{adv} = 1e{-}4$	0.773	0.645	0.807	0.024
Edge gaps:				
$\lambda_{adv} = 1e{-}1$	0.575	0.513	0.817	0.160
$\lambda_{adv} = 1e{-}2$	**0.789**	**0.656**	**0.612**	0.043
$\lambda_{adv} = 1e{-}3$	0.695	0.604	0.855	**0.019**
$\lambda_{adv} = 1e{-}4$	0.755	0.616	0.929	0.021

influence of the L1 loss; the value of delta was tuned via a grid search of values presented in Table 4. For the CE-GAN, the discriminator was trained using a binary cross entropy (BCE) loss, while the generator utilised a BCE / MSE hybrid. The BCE loss is defined as:

$$l_{bce} = -u_{i,\mathrm{pred}} * \log u_{i,\mathrm{true}} + \left(1 - u_{i,\mathrm{pred}}\right) * \log\left(1 - u_{i,\mathrm{true}}\right). \quad (8)$$

The combined generator loss is given by:

$$l_{gen} = \lambda_{adv} * l_{bce} + (1 - \lambda_{adv}) * l_{mse}. \quad (9)$$

The adversarial ratio λ_{adv} controls the relative importance of the MSE and BCE losses. Following Li et al.,[48] the sensitivity of four different adversarial ratios are tested with results reported in Table 5. Finally, the physics-based gradient loss is defined as[82]:

$$l_{phys} = \lambda_{grad} * l_{grad} + \left(1 - \lambda_{grad}\right) * l_{mse} \quad (10)$$

The gradient ratio λ_{grad} controls the relative importance of the gradient and MSE losses. The gradient loss l_{grad} calculates the MSE between the gradients of the feature and target maps, and is defined in two dimensions (2D) as:

$$l_{grad} = l_{mse}\left(\frac{\partial u_{i,\,true}}{\partial x}, \frac{\partial u_{i,\,pred}}{\partial x}\right) + l_{mse}\left(\frac{\partial u_{i,\,true}}{\partial y}, \frac{\partial u_{i,\,pred}}{\partial y}\right). \quad (11)$$

Here, x and y are the two coordinate directions. A gradient loss function helps to preserve sharp transitions and edges which can be smoothed over when using the MSE. In addition, in the case of turbulent flow data, preservation of the gradients can encourage the model to align with physical quantities such as vorticity.[67]

Data augmentation

One of the key considerations of this work is in how artificial gaps should be introduced into the data to train the models. This can be handled via data augmentation at training time. Three different techniques were investigated in this work: introducing fixed horizontal and vertical edge gaps like the test case (fixed edge), blocks of various sizes and locations (random blocks), and edge gaps of random size and orientation (random edge gaps). Some example random block gaps are shown in Figure 6, where the yellow regions indicate areas where data were removed from the snapshots.

The random edge gaps are constructed by taking four random points along the input image borders, drawing a polygon between the points and masking out pixels that lie outside of the polygon. There can be a maximum of two points on any one edge. This approach ensures that edge gaps are created with random sizes and orientations, to prevent the models from overfitting to specific gap shapes and locations. A maximum percentage of the pixels are allowed to be removed by the mask; the mask is discarded if it removes more pixels than this, and a replacement mask is generated. This upper threshold for the gap sizes is needed to constrain the training process, prevent the inpainting task from becoming overly challenging and reflect more realistic physical scenarios. A histogram showing the proportion of pixels removed for each snapshot in one pass of the training set for 10% gaps is shown in Figure 7. A regular PIV snapshot is shown alongside two snapshots with random edge gaps added in Figure 8.

Results

Training gaps

Firstly, the different artificial gap handling strategies previously described were tested with the UNet model, in order to establish the optimal training pipeline. The results for the four metrics tested are given in Table 6, with separate reports for the central image regions and

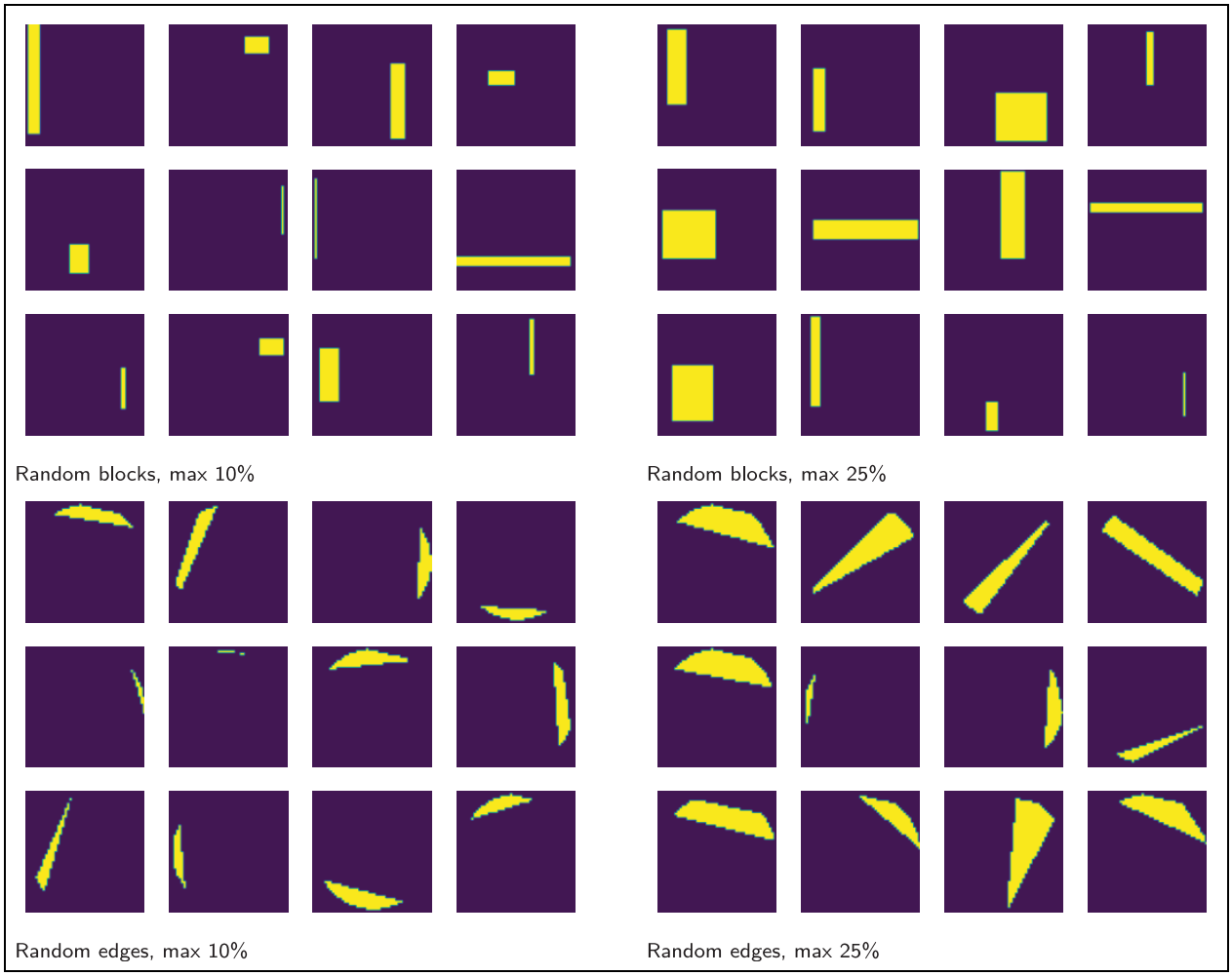

Figure 6. Samples of the randomly-generated block and edge gaps used to train the models in this study, for gap sizes of 10% and 25% of the total area.

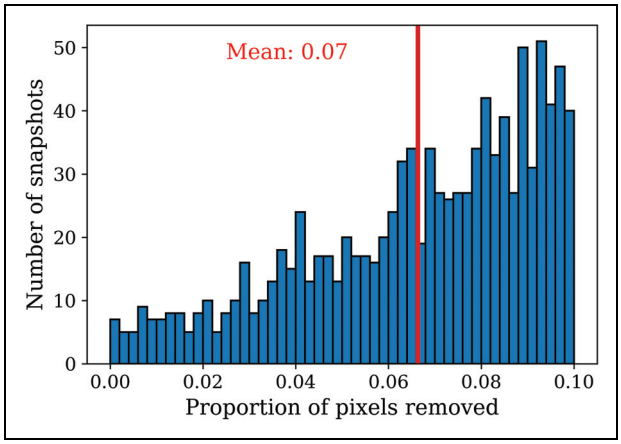

Figure 7. Histogram showing the proportion of pixels removed by the random edge masks in one pass through the training set. Seven percent of the total pixels in the field of view were removed on average.

the gap regions. Overall, the accuracy inside the central regions is very high in all cases, with RI = 0.999 and a pixelwise L2 error of ≈3%. This shows that the UNets are able to preserve the information provided to it in the input to a very high degree, despite compressing the data through the bottleneck. As expected, the accuracy inside the gap regions is worse, as the UNet is required to extrapolate beyond the field of view that was supplied at the input. However, the results still appear to be passable, with RI up to 0.88 at the 10% gap size and 0.82 at 25%.

Training the UNet on fixed edge gaps, which have the same form as the test gaps, generally produced the highest accuracies in the image centres. This simpler training strategy allowed the model to focus more on the global flow patterns provided at the input, as the location of the gaps did not change from image to image. This emphasis on general flow patterns helped to also yield the best KL divergences within the edges at both 10% and 25% gap sizes. However, the weaker RI and L2 scores at the edges indicate that over-fitting the model to the fixed mask shape prevented it from generalising as well to the more specific localised flow behaviour in the unseen crank angle. Conversely, for the random edge gaps, the addition of significant variability to the process made it more challenging for the

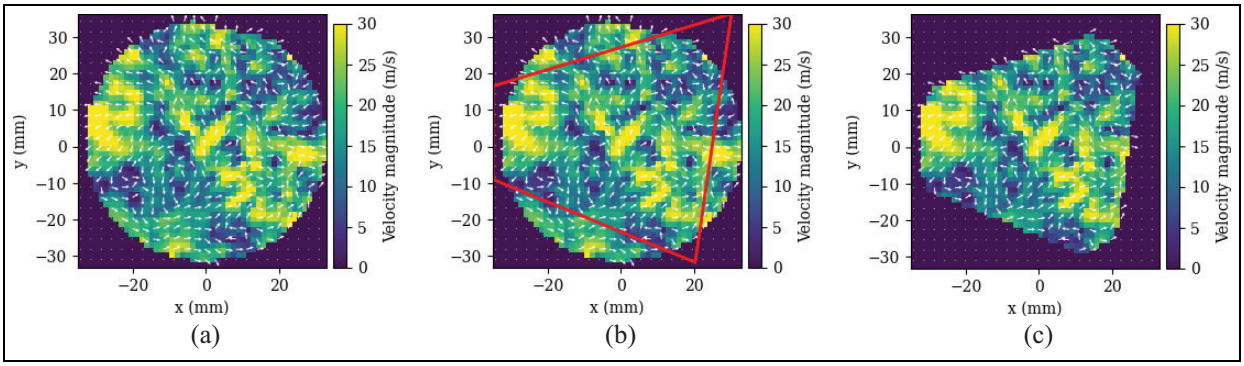

Figure 8. Example random edge gap creation. From left to right: (a) original image, (b) image with a random edge gap polygon superimposed in red and (c) edge gaps added to regions outside of the random polygon.

Table 6. Impact of training a UNet, MSE model on the fixed edges used as the testing gaps, random edge gaps, random block gaps and a combination of the latter two. One result for each setup using permutation A is reported. The final figure represents the average over the 1041 test images. **Bold** typeface represents the best result.

Gap type	Gap size	RI	MI	L2	KL
Central regions:					
Fixed edge	10%	**1.000**	**0.986**	**0.027**	**0.000**
Random edge	10%	**1.000**	0.984	0.032	**0.000**
Random block	10%	**1.000**	0.984	0.031	**0.000**
Random block and edge	10%	0.999	0.983	0.035	**0.000**
Edge gaps:					
Fixed edge	10%	0.878	**0.754**	0.488	**0.009**
Random edge	10%	**0.886**	0.751	**0.463**	0.016
Random block	10%	0.844	0.703	0.528	0.028
Random block and edge	10%	0.879	0.747	0.473	0.010
Central regions:					
Fixed edge	25%	**0.999**	**0.984**	**0.033**	**0.000**
Random edge	25%	**0.999**	0.979	0.042	**0.000**
Random block	25%	**0.999**	0.979	0.043	**0.000**
Random block and edge	25%	**0.999**	0.979	0.043	**0.000**
Edge gaps:					
Fixed edge	25%	0.814	**0.695**	0.609	**0.021**
Random edge	25%	**0.819**	0.688	**0.569**	0.027
Random block	25%	0.803	0.674	0.588	0.035
Random block and edge	25%	0.813	0.683	0.577	0.030

UNet to learn the global flow distributions as precisely. However, the random edge training did improve the predictions of local details and vector orientations, with the strongest RI and L2 scores at the edges.

The random blocks and combination of random blocks and edges were used to test whether a broader inpainting training process would help the model to generalise further. However, neither of these strategies produced higher accuracies than the fixed or random edge gaps in isolation. This shows that for this problem, the best performance can be achieved by providing the model with training and testing gaps that are of the same general shape and location; however, some randomisation within these general parameters via the random edge gaps did provide the strongest RI and L2 metrics inside the edge regions for both gap sizes. In addition, it is expected that models trained on random edge gaps will be able to handle test cases with edge gaps at any orientation, unlike models trained on fixed gap positions. Due to this improved flexibility, combined with strong scores across all four metrics, the random edge gaps method was deemed to have the most practical utility among the data augmentation methods tested here. Therefore, the random edge gaps technique was used in the training pipeline to benchmark the other model configurations investigated in this work.

Loss functions

Use of the Huber, adversarial and gradient loss functions require the tuning of parameters in order to determine suitable configurations. A grid search was performed in each case, following best practices laid out in previous studies.[48,67] The Huber and gradient loss parameters (δ and λ_{adv} respectively) were tuned

Table 7. Gradient loss lambda tuning results for the 180 CAD test case with 25% edge gaps. One result for each setup using permutation A is reported. **Bold** typeface represents the best result.

Parameter	RI	MI	L2	KL
Central regions:				
$\lambda_{grad} = 0.5$	**0.999**	**0.979**	**0.043**	**0.000**
$\lambda_{grad} = 0.9$	**0.999**	0.971	0.059	0.001
$\lambda_{grad} = 0.99$	**0.999**	0.971	0.059	0.001
$\lambda_{grad} = 0.999$	**0.999**	0.969	0.062	0.001
Edge gaps:				
$\lambda_{grad} = 0.5$	0.809	0.687	0.581	**0.024**
$\lambda_{grad} = 0.9$	0.825	0.693	0.559	0.028
$\lambda_{grad} = 0.99$	0.821	0.684	0.566	0.027
$\lambda_{grad} = 0.999$	**0.834**	**0.696**	**0.549**	0.043

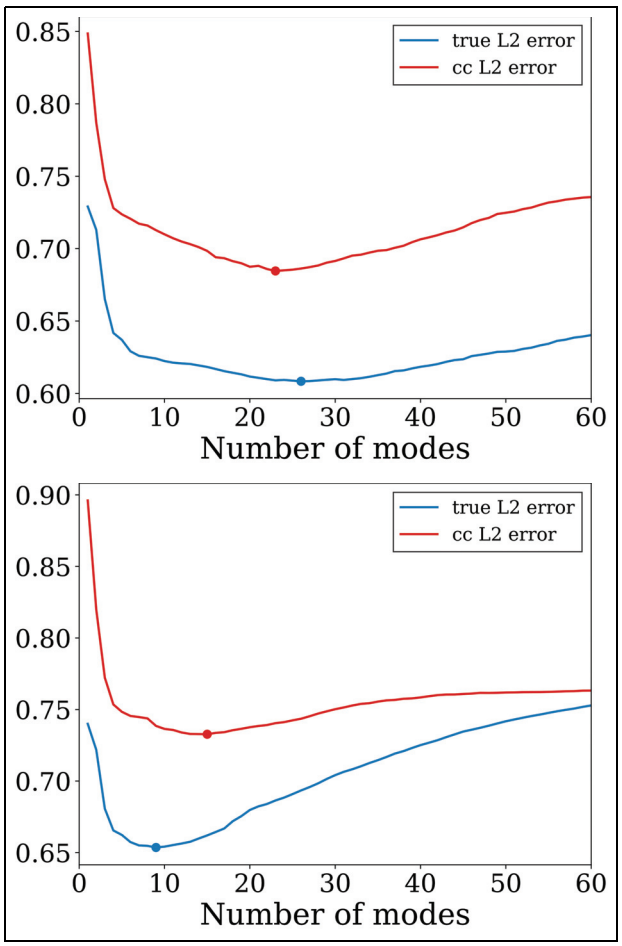

Figure 9. GPOD convergence curves for 10% and 25% gaps (left and right respectively) at permutation A. The minimum errors for both the true L2 error in the edge gaps and the L2 error in the convergence checking (CC) gaps are marked as filled circles.

within a UNet model, while the adversarial ratio λ_{adv} was employed within the CE-GAN. The results are presented in Tables 4 and 5. For the Huber loss, $\delta = 1$ exhibited the best performance in the edge gaps and was therefore used in the remainder of the study. The increased use of the L1 loss at the smaller $\delta = 0.1$ was less sensitive to large errors more likely to be found inside the gaps, but produced the best accuracies in the image centres where the reconstructions are closer to being correct.

For the adversarial ratio, λ_{adv} = 1e–2 yielded the best scores across three of the metrics in the edges and was chosen for the remainder of the CE-GAN results in this study. In this case, λ_{adv} strikes a balance between reconstructing an accurate output with the MSE loss and fooling the discriminator with the BCE loss. The stronger RI, MI and L2 but weaker KL divergence at λ_{adv} = 1e–2 relative to λ_{adv} = 1e–3 indicate that capturing finer details is more useful for overcoming the discriminator than the global velocity distribution. For the gradient loss, $\lambda_{grad} = 0.999$ exhibited the best results, as shown in Table 7, and was used by the UNet, gradient models for the remainder of the study. The λ_{grad} parameter behaves in a similar fashion to the adversarial ratio, with a trade-off between the KL divergence and the RI, MI and L2 scores. In this case, more emphasis on the gradient loss helps to preserve global flow patterns, while the MSE is more effective in predicting local details such as vector orientations and pixelwise values.

GPOD convergence

As the final step before each of the model configurations can be benchmarked againsnt one another, the number of modes to be retained by the GPOD prediction is determined using the convergence criterion. Plots showing the GPOD convergence curves for 10% and 25% gaps in permutation A are shown in Figure 9. In both cases, the relative L2 error calculated in the convergence-checking (CC) gaps gives an optimal number of modes that is relatively close to the true optimum given by the L2 error in the true gaps. The number of modes retained in the final GPOD reconstructions benchmarked here considered the true lowest L2 errors, and were 26 for the 10% gap size and 9 for 25%. The lower number of modes in the latter case indicates that the GPOD algorithm relies on a reconstruction that contains more general flow patterns in order to optimise the error across the larger gaps in the different snapshots.

Main benchmark results

The results for the benchmark performance metrics are given in Tables 8 and 9, with the best result for each metric presented in bold. Loss curves for each model configuration are provided in Figure A1 in Supplemental Material. The UNet and UNETR models exhibit similar performances across all metrics, with the UNet models slightly outperforming UNETR for predictions inside the edge gaps. As shown in Table 2, the number of parameters in the UNet architecture is eight times smaller than that of the UNETR model,

Table 8. Results for the 180 CAD test case at 10% gaps. The mean and standard deviations are reported from the three permutations of training data defined in Table 3. **Bold** typeface represents the best mean in each category separated by the horizontal lines. GPOD-MF metrics are not given in the central regions as adaptive GPOD methods only update values inside the gaps.

Model	Gap size	RI	MI	L2	KL
Central regions:					
UNet, MSE	10%	**1.000 ±0.000**	0.984 ±0.000	0.033 ±0.001	**0.000 ±0.000**
UNet, huber	10%	0.999 ± 0.000	0.983 ±0.000	0.035 ±0.001	**0.000 ±0.000**
UNet, gradient	10%	0.999 ± 0.000	0.974 ±0.001	0.052 ±0.001	0.001 ±0.000
UNETR	10%	**1.000 ±0.000**	**0.985 ±0.001**	**0.030 ±0.001**	**0.000 ±0.000**
CE-GAN	10%	0.884 ±0.003	0.745 ±0.010	0.470 ±0.008	0.019 ±0.004
Edge gaps:					
GPOD-MF	10%	0.797 ±0.001	0.666 ±0.001	0.610 ±0.002	0.105 ±0.001
UNet, MSE	10%	0.890 ±0.004	0.759 ±0.007	0.456 ±0.009	**0.013 ±0.002**
UNet, huber	10%	0.892 ±0.003	**0.760 ±0.003**	0.452 ±0.005	**0.013 ±0.001**
UNet, gradient	10%	**0.894 ±0.002**	0.758 ±0.004	**0.451 ±0.004**	0.016 ±0.001
UNETR	10%	0.884 ±0.002	0.755 ±0.002	0.467 ±0.005	0.014 ±0.001
CE-GAN	10%	0.784 ±0.008	0.661 ±0.010	0.622 ±0.007	0.032 ±0.008

so the UNet exhibits a better accuracy-complexity trade-off. This indicates that detailed local features and textures may be more predictive of the target outputs than global context in this situation, which runs counter to where UNETR models typically see performance gains.[43–45]

The UNet variants each exhibit similar predictive performances in the edge gaps at the 10% gap size, although the gradient loss function demonstrates the best RI, MI and L2 metrics at 25% gaps. This is in line with the results of Chung et al.[67] who showed that the gradient loss provided persisting benefits for a super-resolution task of increasing difficulty from $8\times$ to $32\times$ magnification. On the other hand, in the present work, the UNet, gradient model yields higher KL divergences in the edges, especially at 25% gaps, as shown in Table 9. This shows that the gradient loss function emphasises local regions with large velocity gradients at the expense of the overall energy distribution in the flow. As with the investigation on data augmentation strategies and loss function parameters, this is another example of how the models seem to face something of a trade-off between accurate KL divergences, and RI and L2 errors.

The accuracy of all UNet-based models is very high in the image centres, with KL divergences that round to zero at a three decimal place tolerance, showing that the original flow structures across all scales are being well-preserved. Ensemble averaged energy spectra for the UNet, MSE model predictions at 10% gaps are shown in Figure 10 and there is a near line-on-line match between the true and predicted spectra in the image centres. Note that the energy spectra are challenging to compute in the gappy regions in isolation, as sharp edges and discontinuities are prevalent, contributing to the Gibbs phenomena observed in the edge spectra in Figure 10. However, overall trends can still be seen and the UNet edge prediction follows a downward trend that is similar to the ground truth.

Regarding the other metrics in the edge gaps, the L2 errors of both UNet and UNETR models are relatively high at between 45% and 47%. This is within the range of values reported by Li et al.[48] for large gap sizes, but about twice as high as other results reported by Morimoto et al.[35] for the reconstruction of a turbulent flow in a fixed gap shape. The reasoning behind this is explained in the Discussion section. For the RI and MI values of between 0.9 and –0.95 are commonly taken to represent self-similarity between vector fields.[83] The average RIs for the UNet and UNETR predictions at 10% gap sizes approach this criterion in the edge gaps and meet it in the central regions. The MI values are systematically lower, which is consistent with other reports that the MI is a stricter metric to satisfy, as it follows a linear relationship rather than the sinusoidal RI.[83–85]

Example flow field predictions from the UNet, MSE model at 10% gaps are shown in Figure 11. In the top row of the figure, the regions masked out by the horizontal mask are relatively uniform and easy to predict with no large variations in velocity magnitude. This allows the UNet to predict the flow inside the gaps to a fair degree of accuracy. On the other hand, for the bottom row, turbulent motion inside the edge gap regions is more complex, with the flow directions switching to point outwards just inside the edge gap regions. There are few obvious indicators for this motion in the centre of the image and the UNet struggles to fully predict this complexity. The scarcity of spatially local information due to the edge gaps highlights the challenge presented by this inpainting task; it is likely that more knowledge of the out-of-plane motion would be needed in order to predict such complex behaviour. To provide a clearer picture of the differences between these two flow fields, the point-wise L2 errors are shown in Figure 12. Plots showing example outputs from each of the models at 10% and 25% gap sizes are provided in Figures 13 and 14.

Table 9. Results for the 180 CAD test case at 25% gaps. The mean and standard deviations are reported from the three permutations of training data defined in Table 3. **Bold** typeface represents the best mean in each category separated by the horizontal lines. GPOD-MF metrics are not given in the central regions as adaptive GPOD methods only update values inside the gaps.

Model	Gap size	RI	MI	L2	KL
Central regions:					
UNet, MSE	25%	**0.999 ±0.000**	0.978±0.001	0.044±0.002	**0.000±0.000**
UNet, huber	25%	**0.999 ±0.000**	0.978±0.001	0.045±0.002	**0.000±0.000**
UNet, gradient	25%	**0.999 ±0.000**	0.972±0.003	0.057±0.005	0.001±0.000
UNETR	25%	**0.999 ±0.000**	**0.983±0.001**	**0.034±0.003**	**0.000±0.000**
CE-GAN	25%	0.885 ±0.006	0.739±0.006	0.470±0.011	0.020±0.001
Edge gaps:					
GPOD-MF	25%	0.762 ±0.008	0.629±0.007	0.654±0.009	0.144±0.011
UNet, MSE	25%	0.817 ±0.005	0.691±0.006	0.571±0.006	0.029±0.001
UNet, huber	25%	0.822 ±0.001	0.691±0.004	0.565±0.002	0.028±0.001
UNet, gradient	25%	**0.826 ±0.008**	**0.692±0.003**	**0.559±0.010**	0.040±0.009
UNETR	25%	0.800 ±0.005	0.680±0.004	0.598±0.004	**0.027±0.001**
CE-GAN	25%	0.735 ±0.005	0.620±0.004	0.677±0.005	0.055±0.004

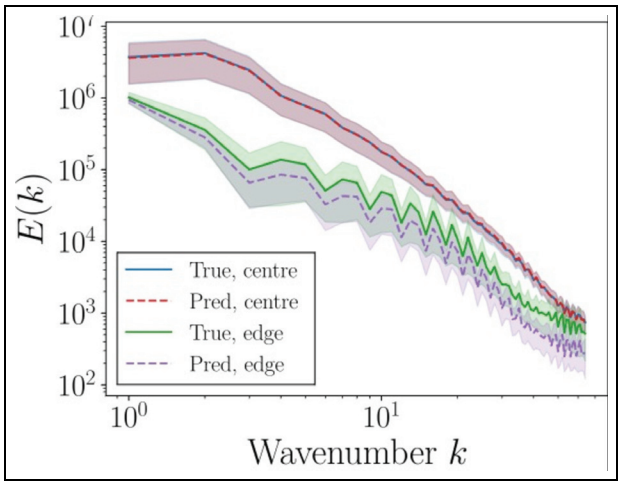

Figure 10. Energy spectra comparing the ground truth test set images to the UNet, MSE predictions at a 10% test gap size. Ensemble mean spectra are given by solid or dashed lines. Shaded areas represent one standard deviation from the mean.

The CE-GAN demonstrates relatively poorer performance across the board. Li et al.[48] also reported relatively low pixel-wise accuracies for the CE-GAN in an inpainting task on PIV data, but better performance than GPOD in terms of predicting multi-scale properties. These findings are supported in the present study; however, Tables 8 and 9 show that the CE-GAN results are worse than UNet-based models across all metrics, especially in the central image regions. Finally, the GPOD-MF method yields the lowest performance, as GPOD-MF is sensitive to the limited amount of spatial information available near the gap regions. Difficulties arise because the algorithm initialises the gaps with ensemble mean vectors calculated from the training set, then iterates on these guesses using the dominant flow features from POD-based reconstructions. However, for highly variational flows, the mean can be a poor approximation of the full dataset.[84] GPOD-MF can typically overcome this by utilising the local spatial information to update the guesses, but this is not so effective for large block gaps and errors can be compounded instead. Figure 9 shows that the algorithm converges at a relatively small number of modes, representing the dominant flow structures. While these dominant structures do not fare as badly on the global vector-based metrics, they are overly smoothed and differ vastly in terms smaller-scale flow structures, as shown by the large KL divergences between the GPOD-MF predictions and the true vectors in the edge gaps.

Discussion

These results have shown that UNet-based models are capable of extrapolating beyond the field of view by reconstructing the flow inside edge gaps to a reasonable degree of accuracy, significantly out-performing GPOD. At 10% gap size, all three UNets achieved an RI of at least 0.89 on average for the unseen crank angle, showing that vector alignments can be well-predicted in general. However, the 25% gap size presents a much harder challenge, with RIs falling to ~0.82. The raw metrics at 10% and 25% gap sizes might not seem too disparate at first, but a visual inspection of Figures 13 and 14 reveals that the differences between these scores has significant consequences in the predicted flow fields. While the predicted flow fields at the 10% gap size appear to be reasonable in general, the predictions at the 25% gap size are unreliable, with large inaccuracies in the predicted flow motion.

This poses an interesting question as to what level of accuracy should be expected from an inpainting model and whether it is possible to accurately predict the flow inside edge gaps as large as 25% of the total pixels. The process of inpainting in this case relies on there being a

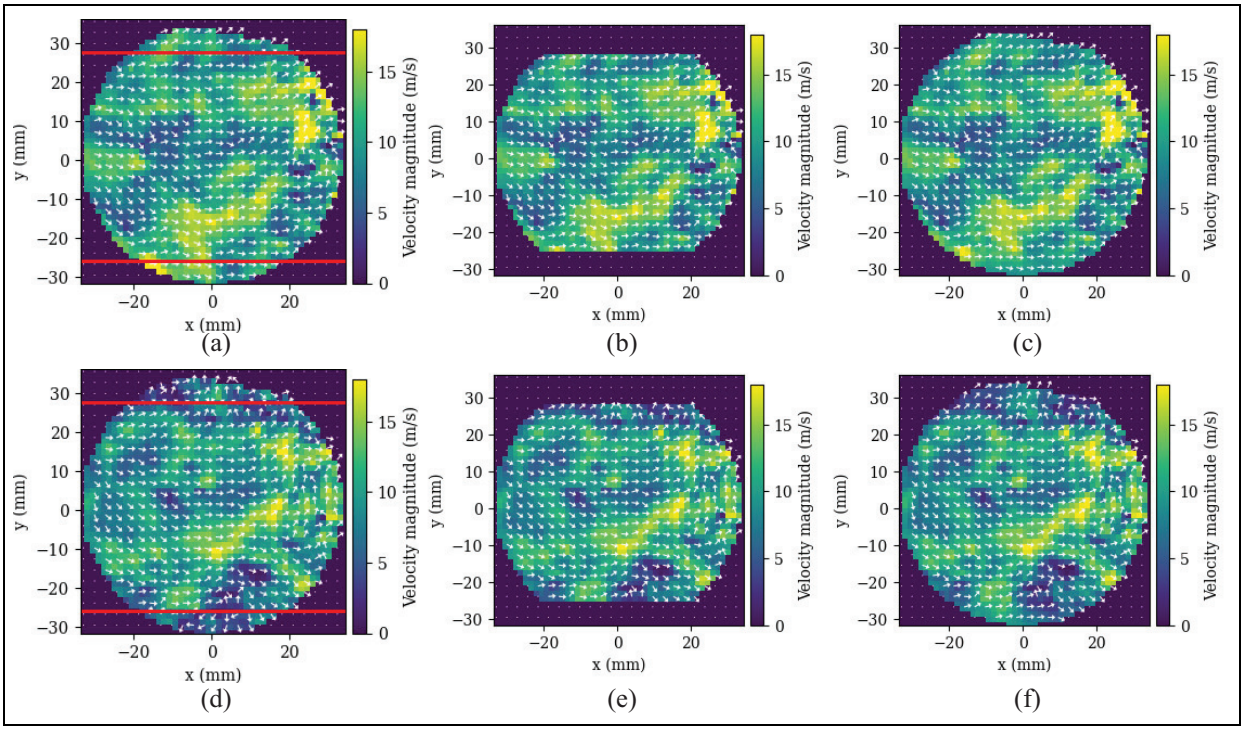

Figure 11. Sample flow fields from the 10% gaps test set. Top row: best UNet, MSE prediction (L2 = 0.225); bottom row: worst UNet, MSE prediction (L2 = 1.026). (a and d) Original snapshot with the test mask shown as horizontal red lines, (b and e) gappy input and (c and f) prediction.

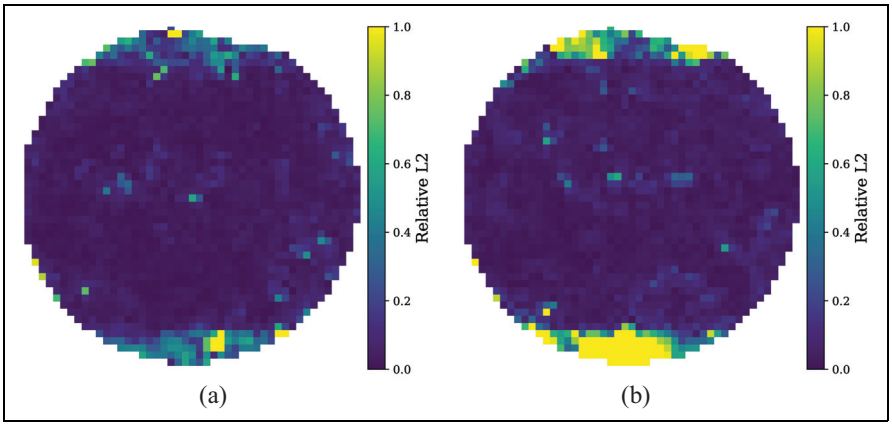

Figure 12. Pixel-wise L2 errors between the UNet predictions and the original images for (a) the best (top row of Figure 11) and (b) the worst predictions (bottom row of Figure 11).

strong correlation between the flow at the centre of image and the flow inside edge gaps. For the larger gap size, it is less likely for such a correlation to exist, as the distance between the outer gap regions and the nearest data-containing pixel is increased. If there is no strong correlation between these different regions of the flow, then this becomes an ill-posed problem, with many possible options for the flow behaviour inside the edge gaps.[79] Future work should investigate how the internal correlations within the flow relate to an inpainting model's performance, to further inform what is and is not possible within this task.

Comparisons between the neural network models here show that UNet-based models exhibit similar performances, while the CE-GAN accuracies are markedly worse. In particular, the low accuracies inside the central regions indicate that the CE-GAN is not retaining as much of the information in the image centres as the UNet-based models are. Indeed, although both models incorporate autoencoder-like structures, while the CE-GAN generator has an AlexNet-like architecture,[64] UNet-based models utilise skip connections that are designed to preserve contextual information at each stage of the autoencoder. This allows the UNets to

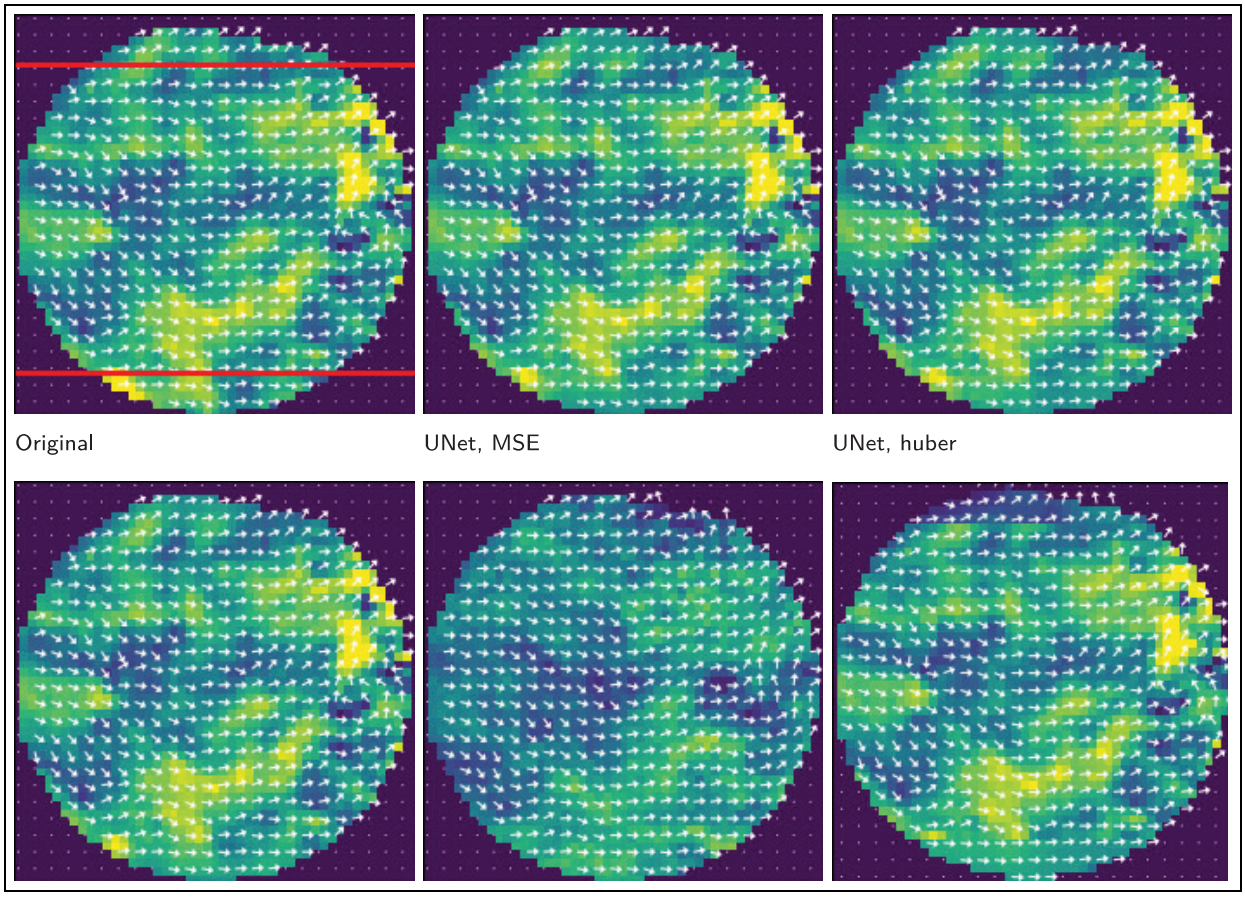

Figure 13. Comparison of different model predictions for a single test snapshot at 10% gaps. Gappy images formed by removing data outside of the red lines in the original image are fed into the models.

simultaneously preserve information in the image centre to a very high degree of accuracy and yield gap predictions that seamlessly integrate with the rest of the image. This has also correlated with better edge gap predictions for the UNet-based models in this case. It should be noted that this particular application pushes the CE-GAN beyond its initial design intention of solely predicting inside the gap region, rather than also reproducing the entire image. The other key difference between the UNets and the CE-GAN is the adversarial training dynamic employed by the latter, which if unstable, could also contribute to lower performances.

A note is needed regarding the relatively high L2 errors, of between 45% and 47% for the UNet-based models. It is hypothesised that the main reason for the higher L2 errors reported in this benchmark is the significant out-of-plane motion present in the TCC-III engine, which is not accounted for currently. As shown in the bottom row of Figure 11, the flow at the image centres reveal few indicators of the sudden change in vector directions inside the edge gaps. However, only two velocity components along a single PIV plane are observed here. It is possible that accounting for out-of-plane motion by gaining access to the third velocity component using techniques such as tomographic PIV[86] or assimilation with CFD data[87] will be required to significantly reduce the L2 errors in this situation and such investigations will constitute future work.

With the present results as they are, it is recommended that UNet-based models can be used to reconstruct the flow in large block gaps with as many as 10% of the total number of pixels missing to a reasonable degree of accuracy. Such reconstructed flow fields could be used to improve understanding of general flow patterns and replace ensemble mean-filled or interpolated flow fields as inputs into other data analysis methods like modal decomposition. However, despite the well-predicted vector alignments, care should be taken when using the vector magnitudes from the predicted flow patterns, as these were found to under-predict the ground-truth values. Note that the performance of the UNets is expected to improve for easier inpainting tasks such as smaller and more centrally-located blocks of missing data, in which case the prediction of these vector magnitudes would likely improve. Testing the sensitivity of inpainting models to a range of gap types is also recommended for future work. Finally, reliably inpainting edge gaps for the 25% gap size appears to be out of reach at present and as previously discussed, an investigation into how feasible it would be for any model to accurately reconstruct the flow in such scenarios is recommended future work.

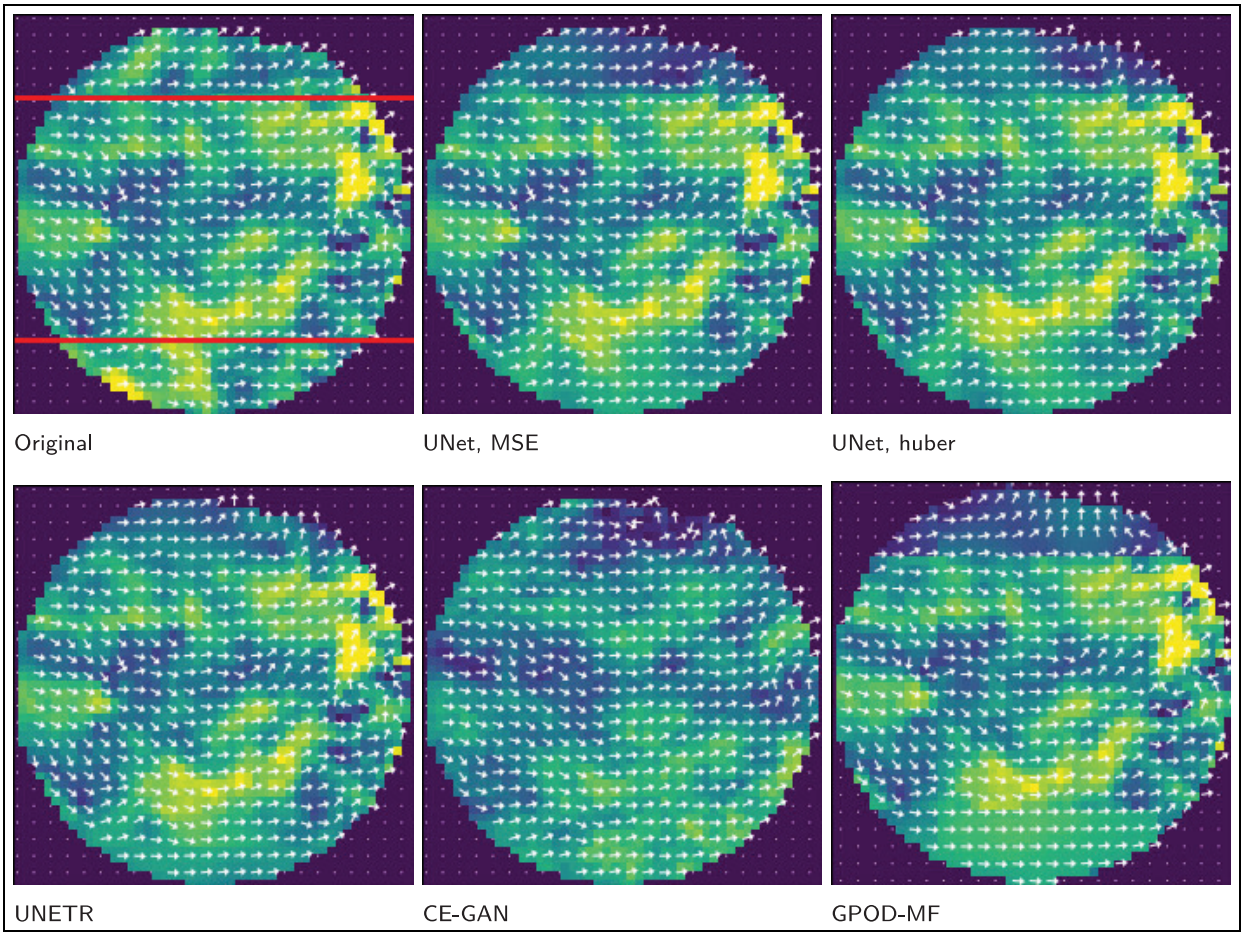

Figure 14. Comparison of different model predictions for a single test snapshot at 25% gaps. Gappy images formed by removing data outside of the red lines in the original image are fed into the models.

Conclusion

This work has introduced the EngineBench database and used it to establish the first inpainting benchmark for an industrially-relevant turbulent flow, in order to address the limited availability of practical benchmarks on experimental data. The models were tasked with inpainting large edge gaps, a highly challenging problem that pushes the models to the limits of their capabilities. This benchmark was used to provide objective insight into how a range of widely-used models behave in these challenging conditions, identify success and failure modes and provide recommendations for future work.

Firstly, a number of data augmentation strategies that introduce artifical gaps of different forms into the data are tested to find the optimum strategy for inpainting edge gaps. A novel strategy, named random edge gaps, was created to introduce edge gaps of random sizes, locations and orientations into the training data. Although fixed gap training yielded the best results in terms of MI and KL divergence, random edge gap training was shown to result in the most accurate predictions of vector orientations and pixel-wise errors due to the improved generalisability. Random edge gap training is therefore recommended for the creation of flexible and generalisable inpainting models in this case.

Overall, UNet-based models demonstrated the best general performance across the four metrics and two gap sizes tested. Indeed, the UNet-based model predictions in the edge gaps approached self-similarity at the 10% gap size according to the vector-based metrics. This suggests that even gaps as challenging as large edge gaps can be reconstructed to a reasonable degree of accuracy with the use of a UNet, which exhibits significant performance improvements over the GPOD method for this type of gap. However, pixel-wise L2 errors remained relatively high for all model predictions. A visual inspection of the reconstructed flow fields revealed that sudden changes in the flow direction without any obvious indicators in the rest of the flow was a cause of lower reconstruction accuracies. It is therefore hypothesised that acquiring information on the out-of-plane motion, such as through stereo-PIV or data assimilation with CFD, would be needed in order to rectify this issue.

A comparison between the UNet-based models and the CE-GAN showed that the former were capable of preserving the central flow information provided at the input to a very high degree of accuracy and then

leveraging this information to produce better predictions inside the edge gaps. This suggests that skip-connections, contained within UNet-based models but not the CE-GAN generator, are important architectural components that facilitate high accuracies for neural networks training to reconstruct turbulent flow data. This characteristic should be considered in future model development.

In summary, the main practical findings arising from this article are that deep learning methods can outperform GPOD for challenging flow reconstruction tasks; the performance of UNet-based models for up to 10% relative gap sizes is likely sufficient for informing general flow patterns and contributing to data analysis approaches such as modal decomposition methods, but not predicting detailed physical phenomena; the random edge gaps data augmentation technique is effective in helping the deep learning models to generalise and reach higher performances. Recommended future work consists of investigating methods of incorporating information on the out-of-plane motion into the inpainting task, exploring how the spatial correlations within the flow impact expected inpainting performance and testing inpainting models on a variety of different gap types including random noise and other forms of block gaps.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship and/or publication of this article: This research was funded in whole or in part by the Engineering and Physical Sciences Research Council Prosperity Partnership (EPSRC), Grant No. EP/T005327/1. For the purpose of Open Access, the author has applied a CC BY public copyright license to any Author Accepted Manuscript (AAM) version arising from this submission. The Prosperity Partnership is a collaboration among Jaguar Land Rover, Siemens Digital Industries Software, the University of Bath and the University of Oxford. Samuel Baker acknowledges support from EPSRC, Grant No. EP/T518232/1.

ORCID iDs

Samuel Baker https://orcid.org/0000-0002-4470-2122
XiaoHang Fang https://orcid.org/0000-0001-6360-9065
Felix Leach https://orcid.org/0000-0001-6656-2389
Martin Davy https://orcid.org/0000-0001-7866-9028

References

1. Schiffmann P, Gupta S, Reuss D, etal. TCC-III engine bench mark for large-Eddy simulation of ICengine flows. *Oil Gas Sci Technol* 2016; 71(1): 3.
2. Greitzer EM, Tan CS and Graf MB. *Internal flow: concepts and applications*. Cambridge: Cambridge University Press, 2007.
3. Kleinstreuer C and Zhang Z. Airflow and particle transport in the human respiratory system. *Annu Rev Fluid Mech* 2010; 42(1): 301–334.
4. Hasler D, Landolt A and Obrist D. Tomographic piv behind a prosthetic heart valve. *Exp Fluids* 2016; 57: 1–13.
5. Tomasini E, Paone N, Rossi M, et al. Overview on piv application to appliances. *Part Image Velocim* 2008; 32: 271–281.
6. Chen XX, Tzeng SJ and Wang WC. Numerical and experimental observations of the flow field inside a selective laser melting (SLM) chamber through computational fluid dynamics (CFD) and particle image velocimetry (PIV). *Powder Technol* 2020; 362: 450–461.
7. Senecal K and Leach F. *Racing toward zero: the untold story of driving green*. Warrendale, PA: SAE International, 2021.
8. Liu Y, Sun X, Sethi V, et al. Review of modern low emissions combustion technologies for aero gas turbine engines. *Prog Aerosp Sci* 2017; 94: 12–45.
9. Adrian RJ and Westerweel J. *Particle image velocimetry*. 30th ed. Cambridge: Cambridge University Press, 2011.
10. Scherl I, Strom B, Shang JK, et al. Robust principal component analysis for modal decomposition of corrupt fluid flows. *Phys Rev Fluids* 2020; 5(5): 054401.
11. Van Doorne C and Westerweel J. Measurement of laminar, transitional and turbulent pipe flow using stereoscopic-piv. *Exp Fluids* 2007; 42: 259–279.
12. Chandramouli P, Mémin E and Heitz D. 4d large scale variational data assimilation of a turbulent flow with a dynamics error model. *J Comput Phys* 2020; 412: 109446.
13. Anderson JD and Wendt J. *Computational fluid dynamics*, vol. 206. New York, NY: Springer, 1995.
14. Aversano G, Ferrarotti M and Parente A. Digital twin of a combustion furnace operating in flameless conditions: reduced-order model development from CFD simulations. *Proc Combust Inst* 2021; 38(4): 5373–5381.
15. Brunton SL, Nathan Kutz J, Manohar K, et al. Data-driven aerospace engineering: reframing the industry with machine learning. *AIAA J* 2021; 59(8): 2820–2847.
16. Argyropoulos CD and Markatos N. Recent advances on the numerical modelling of turbulent flows. *Appl Math Model* 2015; 39(2): 693–732.
17. Yousif MZ, Yu L, Hoyas S, et al. A deep-learning approach for reconstructing 3d turbulent flows from 2d observation data. *Sci Rep* 2023; 13(1): 2529.
18. Dubois P, Gomez T, Planckaert L, et al. Machine learning for fluid flow reconstruction from limited measurements. *J Comput Phys* 2022; 448: 110733.
19. Fukami K, Fukagata K and Taira K. Machine-learning-based spatio-temporal super resolution reconstruction of turbulent flows. *J Fluid Mech* 2021; 909: A9.
20. Everson R and Sirovich L. Karhunen–loeve procedure for gappy data. *JOSA A* 1995; 12(8): 1657–1664.
21. Gunes H, Sirisup S and Karniadakis GE. Gappy data: to krig or not to krig? *J Comput Phys* 2006; 212(1): 358–382.
22. Raben SG, Charonko JJ and Vlachos PP. Adaptive gappy proper orthogonal decomposition for particle image velocimetry data reconstruction. *Meas Sci Technol* 2012; 23(2): 025303.

23. Saini P, Arndt CM and Steinberg AM. Development and evaluation of gappy-pod as a data reconstruction technique for noisy piv measurements in gas turbine combustors. *Exp Fluids* 2016; 57: 1–15.
24. Nekkanti A and Schmidt OT. Gappy spectral proper orthogonal decomposition. *J Comput Phys* 2023; 478: 111950.
25. Murata T, Fukami K and Fukagata K. Nonlinear mode decomposition with convolutional neural networks for fluid dynamics. *J Fluid Mech* 2020; 882: A13.
26. Baker S, Fang X, Barbato A, et al. Extracting vector magnitudes of dominant structures in a cyclic engine flow with dimensionality reduction. *Phys Fluids* 2024; 36(2): 12–32.
27. Taira K, Brunton SL, Dawson ST, et al. Modal analysis of fluid flows: an overview. *AIAA J* 2017; 55(12): 4013–4041.
28. Epps BP and Krivitzky EM. Singular value decomposition of noisy data: noise filtering. *Exp Fluids* 2019; 60(8): 1–23.
29. Roudnitzky S, Druault P and Guibert P. Proper orthogonal decomposition of in-cylinder engine flow into mean component, coherent structures and random gaussian fluctuations. *J Turb* 2006; 12(7): N70.
30. Elharrouss O, Almaadeed N, Al-Maadeed S, et al. Image inpainting: a review. *Neural Process Lett* 2020; 51: 2007–2028.
31. Jam J, Kendrick C, Walker K, et al. A comprehensive review of past and present image inpainting methods. *Comput Vis Image Underst* 2021; 203: 103147.
32. Kumar Y, Bahl P and Chakraborty S. State estimation with limited sensors–a deep learning based approach. *J Comput Phys* 2022; 457: 111081.
33. Nguyen T, Jewik J, Bansal H, et al. Climatelearn: benchmarking machine learning for weather and climate modeling. *Adv Neural Inf Process Syst* 2024; 36: 32–43.
34. Jin X, Cheng P, Chen WL, et al. Prediction model of velocity field around circular cylinder over various reynolds numbers by fusion convolutional neural networks based on pressure on the cylinder. *Phys Fluids* 2018; 30(4): 22–34.
35. Morimoto M, Fukami K and Fukagata K. Experimental velocity data estimation for imperfect particle images using machine learning. *Phys Fluids* 2021; 33(8): 14–28.
36. Ronneberger O, Fischer P and Brox T. U-Net: convolutional networks for biomedical image segmentation. In: *Medical image computing and computer-assisted intervention – MICCAI 2015: 18th international conference*, Munich, Germany, October 5–9, 2015, proceedings, part III 18, pp.234–241. New York, NY: Springer.
37. Kaltenborn J, Lange C, Ramesh V, et al. ClimateSet: a large-scale climate model dataset for machine learning. *Adv Neural Inf Process Syst* 2023; 36: 21757–21792.
38. Bao K, Zhang X, Peng W, et al. Deep learning method for super-resolution reconstruction of the spatio-temporal flow field. *Adv Aerodyn* 2023; 5(1): 19.
39. Zhang J, Liu J and Huang Z. Improved deep learning method for accurate flow field reconstruction from sparse data. *Ocean Eng* 2023; 280: 114902.
40. Deng Z, Liu H, Shi B, et al. Temporal predictions of periodic flows using a mesh transformation and deep learning-based strategy. *Aerosp Sci Technol* 2023; 134: 108081.
41. Ashkboos S, Huang L, Dryden N, et al. ENS-10: a dataset for post-processing ensemble weather forecasts. *Adv Neural Inf Process Syst* 2022; 35: 21974–21987.
42. Wang X, Girshick R, Gupta A, et al. Non-local neural networks. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, IEEE, 2018, Salt Lake City, pp.7794–7803.
43. Kang H, Kim Y, Le TTH, et al. A new fluid flow approximation method using a vision transformer and a u-shaped convolutional neural network. *AIP Adv* 2023; 13(2): 112–134.
44. Jiang J, Li G, Jiang Y, et al. TransCFD: a transformer-based decoder for flow field prediction. *Eng Appl Artif Intell* 2023; 123: 106340.
45. Xu Y, Sha Y, Wang C, et al. Estimation of cavitation velocity fields based on limited pressure data through improved U-shaped neural network. *Phys Fluids* 2023; 35(8): 62–82.
46. Güemes A, Sanmiguel Vila C and Discetti S. Super-resolution generative adversarial networks of randomly-seeded fields. *Nat Mach Intell* 2022; 4(12): 1165–1173.
47. Kim H, Kim J, Won S, et al. Unsupervised deep learning for super-resolution reconstruction of turbulence. *J Fluid Mech* 2021; 910: A29.
48. Li T, Buzzicotti M, Biferale L, et al. Multi-scale reconstruction of turbulent rotating flows with proper orthogonal decomposition and generative adversarial networks. *J Fluid Mech* 2023; 971: A3.
49. Raissi M, Perdikaris P and Karniadakis GE. Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J Comput Phys* 2019; 378: 686–707.
50. Krishnapriyan A, Gholami A, Zhe S, et al. Characterizing possible failure modes in physics-informed neural networks. *Adv Neural Inf Process Syst* 2021; 34: 26548–26560.
51. Eivazi H, Tahani M, Schlatter P, et al. Physics-informed neural networks for solving reynolds-averaged navier–stokes equations. *Phys Fluids* 2022; 34(7): 127–135.
52. Wang R, Kashinath K, Mustafa M, et al. Towards physics-informed deep learning for turbulent flow prediction. In: *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. ACM, 2020, pp.1457–1466.
53. Wang H, Liu Y and Wang S. Dense velocity reconstruction from particle image velocimetry/particle tracking velocimetry using a physics-informed neural network. *Phys Fluids* 2022; 34(1): 52–68.
54. Kipf TN and Welling M. Semi-supervised classification with graph convolutional networks, arXiv preprint arXiv:1609.02907, 2016.
55. Pfaff T, Fortunato M, Sanchez-Gonzalez A, et al. Learning mesh-based simulation with graph networks. International Conference on Learning Representations, ICLR, Vienna, Austria, 2021, 2020.
56. Gao H and Ji S. Graph u-nets. In: *International conference on machine learning*. PMLR, pp.2083–2092.
57. Lu L, Jin P and Karniadakis GE. DeepONet: learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators, arXiv preprint arXiv:1910.03193, 2019.

58. Kovachki N, Li Z, Liu B, et al. Neural operator: learning maps between function spaces with applications to pdes. *J Mach Learn Res* 2023; 24(89): 1–97.
59. Li Z, Kovachki N, Azizzadenesheli K, et al. Fourier neural operator for parametric partial differential equations, arXiv preprint arXiv:2010.08895, 2020.
60. Venturi D and Karniadakis GE. Gappy data and reconstruction procedures for flow past a cylinder. *J Fluid Mech* 2004; 519: 315–336.
61. Wang H, Gao Q, Feng L, et al. Proper orthogonal decomposition based outlier correction for piv data. *Exp Fluids* 2015; 56: 1–15.
62. Luo Z, Wang L, Xu J, et al. Reconstruction of missing flow field from imperfect turbulent flows by machine learning. *Phys Fluids* 2023; 35(8): 56–78.
63. Buzzicotti M, Bonaccorso F, Di Leoni PC, et al. Reconstruction of turbulent data with deep generative models for semantic inpainting from turb-rot database. *Phys Rev Fluids* 2021; 6(5): 050503.
64. Krizhevsky A, Sutskever I and Hinton GE. ImageNet classification with deep convolutional neural networks. *Adv Neural Inf Process Syst* 2012; 25(2): 5386.
65. Deng J, Dong W, Socher R, et al. ImageNet: a large-scale hierarchical image database. In: *2009 IEEE conference on computer vision and pattern recognition*, 2009, pp.248–255. New York, NY: IEEE.
66. Li Y, Perlman E, Wan M, et al. A public turbulence database cluster and applications to study lagrangian evolution of velocity increments in turbulence. *J Turb* 2008; 34(9): N31.
67. Chung WT, Akoush B, Sharma P, et al. Turbulence in focus: benchmarking scaling behavior of 3d volumetric super-resolution with blastnet 2.0 data. *Adv Neural Inf Process Syst* 2024; 36: 132–148.
68. McConkey R, Yee E and Lien FS. A curated dataset for data-driven turbulence modelling. *Sci Data* 2021; 8(1): 255.
69. Bonnet F, Mazari J, Cinnella P, et al. Airfrans: high fidelity computational fluid dynamics dataset for approximating Reynolds-Averaged Navier–Stokes solutions. *Adv Neural Inf Process Syst* 2022; 35: 23463–23478.
70. Zhou R, Balusamy S, Sweeney MS, et al. Flow field measurements of a series of turbulent premixed and stratified methane/air flames. *Combust Flame* 2013; 160(10): 2017–2028.
71. Meijer M, Somers B, Johnson J, et al. Engine combustion network (ECN): characterization and comparison of boundary conditions for different combustion vessels. *Atom Sprays* 2012; 22(9): 124–135.
72. Ko I, Rulli F, Fontanesi S, et al. Methodology for the large-Eddy simulation and particle image velocimetry analysis of large-scale flow structures on TCC-III engine under motored condition. *Int J Engine Res* 2021; 22(8): 2709–2731.
73. Chen H, Reuss DL and Sick V. On the use and interpretation of proper orthogonal decomposition of in-cylinder engine flows. *Meas Sci Technol* 2012; 23(8): 085302.
74. Chen H, Reuss DL, Hung DL, et al. A practical guide for using proper orthogonal decomposition in engine research. *Int J Engine Res* 2013; 14(4): 307–319.
75. Rabault J, Vernet JA, Lindgren B, et al. A study using piv of the intake flow in a diesel engine cylinder. *Int J Heat Fluid Flow* 2016; 62: 56–67.
76. Petersen B and Miles P. PIV measurements in the swirl-plane of a motored light-duty diesel engine. *SAE Int J Engines* 2011; 4(1): 1623–1641.
77. Hatamizadeh A, Tang Y, Nath V, et al. UNETR: transformers for 3D medical image segmentation. In: *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, IEEE, Waikoloa, 2022, pp.574–584.
78. Cardoso MJ, Li W, Brown R, et al. MONAI: an open-source framework for deep learning in healthcare. arXiv preprint arXiv:2211.02701, 2022.
79. Pathak D, Krahenbuhl P, Donahue J, et al. Context encoders: feature learning by inpainting. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, IEEE, Las Vegas, 2016, pp.2536–2544.
80. Liu H, Wan Z, Huang W, et al. PD-GAN: probabilistic diverse gan for image inpainting. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, IEEE, 2021, pp.9371–9381.
81. Hu B, Banerjee S, Liu K, et al. Large eddy simulation of a turbulent non-reacting spray jet. In: *Internal combustion engine division fall technical conference*, vol. 57281, p.V002T06A007. American Society of Mechanical Engineers. ASME, Houston, 2015.
82. Yu J, Lu L, Meng X, et al. Gradient-enhanced physics-informed neural networks for forward and inverse pde problems. *Comput Methods Appl Mech Eng* 2022; 393: 114823.
83. Rulli F, Fontanesi S, d'Adamo A, et al. A critical review of flow field analysis methods involving proper orthogonal decomposition and quadruple proper orthogonal decomposition for internal combustion engines. *Int J Engine Res* 2021; 22(1): 222–242.
84. Baker S, Fang X, Shen L, et al. Dynamic mode decomposition for the comparison of engine in-cylinder flow fields from particle image velocimetry (PIV) and Reynolds-Averaged Navier–Stokes (RANS) simulations. *Flow Turb Combust* 2023; 111(1): 115–140.
85. Barbato A, Iacovano C and Fontanesi S. Cold-flow investigation of the darmstadt engine with focus on statistical convergence: experimental and large eddy simulation analysis. *Flow Turb Combust* 2023; 110(1): 59–89.
86. Hill H, Ding CP, Baum E, et al. An application of tomographic PIV to investigate the spray-induced turbulence in a direct-injection engine. *Int J Multiph Flow* 2019; 121: 103116.
87. Donato L, Galletti C and Parente A. Self-updating digital twin of a hydrogen-powered furnace using data assimilation. *Appl Therm Eng* 2024; 236: 121431.

Appendix A: Training Loss Curves

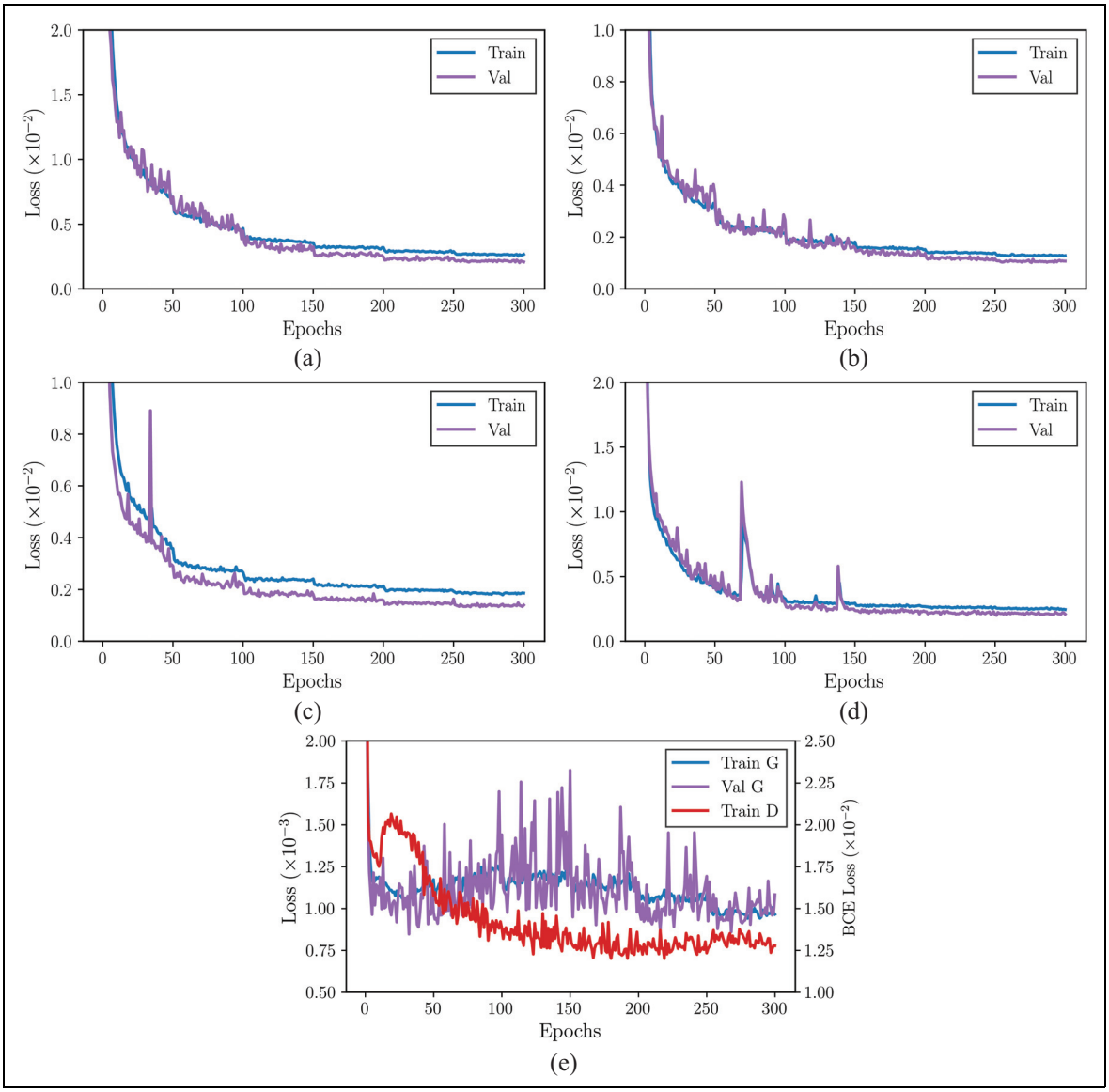

Figure A1. Training loss curves for each of the parametric models for permuation A at 10% gaps. In the CE-GAN plot in (e), note that the generator 'G' was trained using a combined MSE and BCE loss, while the discriminator 'D' was trained solely with the BCE loss. (a) UNet, MSE. (b) UNet, Huber. (c) UNet, gradient. (d) UNETR. (e) CE-GAN.