Benchmarking machine learning models for predicting aerofoil performance

Oliver Summerell, Gerardo Aragon-Camarasa and Stephanie Ordonez Sanchez

Abstract—This paper investigates the capability of Neural Networks (NNs) as alternatives to the traditional methods to analyse the performance of aerofoils used in the wind and tidal energy industry. The current methods used to assess the characteristic lift and drag coefficients include Computational Fluid Dynamics (CFD), thin aerofoil and panel methods, all face trade-offs between computational speed and the accuracy of the results and as such NNs have been investigated as an alternative with the aim that it would perform both quickly and accurately. As such, this paper provides a benchmark for the windAI_bench dataset [1] published by the National Renewable Energy Laboratory (NREL) in the USA. In order to validate the methodology of the benchmarking, the AirfRANS dataset [2] is used as both a starting point and a point of comparison. This study evaluates four neural networks (MLP, PointNet, GraphSAGE, GUNet) trained on a range aerofoils at 25 angles of attack (4° to 20°). to predict fluid flow and calculate lift coefficients (C_L) via the panel method. GraphSAGE and GUNet performed well during the testing phase, but underperformed during validation. Accordingly, this paper has identified PointNet and MLP as the two strongest models tested, however whilst the results from MLP are more commonly correct for predicting the behaviour of the fluid, the results from PointNet provide the more accurate results for calculating C_L .

I. INTRODUCTION

In an age of rising energy demands, tidal stream technologies have the potential to supply a significant level of clean and renewable energy, with the UK alone having an estimated, practical resource of 34TWh/year [3]. Whilst this is equivalent to around 11% of the national energy consumption of the UK, very little of it has been realised at the point of writing [3].

One of the key reasons for the lack of tidal energy being tapped in the UK is the cost attributed to the design and installation of the turbines. The Contracts for Difference scheme, a UK government scheme for the support of low-carbon energy production, priced tidal stream energy at £172/MWh in 2024, compared to offshore wind, which costs £58.87/MWh or £139.93/MWh for floating offshore [4].

 $\@$ 2025 European Wave and Tidal Energy Conference. This paper has been subjected to single-blind peer review.

This work was supported in part by the EPSRC under grant 2890149

O. Summerell and G. Aragon-Camarasa are with the CVAS Group in the Computer Science Dept. of the University of Glasgow, University Avenue, Glasgow, G12 8QQ (email: o.summerell.1@research.gla.ac.uk)

S. Ordonez Sanchez is with the Mechanical and Aerospace Engineering Dept. of Strathclyde University, 16 Richmond Street, Glasgow, G1 1XQ (email: s.ordonez@strath.ac.uk)

Digital Object Identifier:

https://doi.org/10.36688/ewtec-2025-879

The process of designing a tidal turbine requires a variety of methods to evaluate and optimise their operation. The most commonly used methods include a variety of versions of Computational Fluid Dynamics (CFD). Faster and less computationally expensive methods include the use of Blade Element Momentum theory (BEMT) [5]. BEMT, however, relies on a number of correction factors and databases to account for the aero/hydrodynamics of the aerofoil distribution along the blade [6]. XFoil is often used to obtain values, as it provides a balance between computational speed and accuracy. As shown in Morgado et al. [7], XFoil runs quickly and accurately, but it experiences a dramatic drop in accuracy when subjected to high Re or high angles of attack compared to CFD methods. Furthermore, XFoil is unable to 'account for transition with Navier-Stokes solvers' [7], reducing its usefulness for more complex use cases.

As a result, the last few years have seen Neural Networks (NNs) being proposed as a potential alternative to the current methods, with the aim to provide comparable results to CFD models whilst requiring lower computational cost [8]. One of the main drawbacks of using NNs is the reliance on large quantities of reliable data to be trained on, and due to the computational intensity of CFD simulations, few relevant datasets are available. That is to say, whilst there are a variety of fluid flow datasets available, such as CFDBench [8] and MegaFlow2D [9], they do not contain data for aerofoils and, therefore cannot be used for the purposes of aerofoil performance prediction.

There are two datasets being considered within this paper, firstly is windAI_bench [1], published by NREL, which contains a large number of aerofoils, more than sufficient for use in NNs. The actual layout of this data is explored in section III-A. To the knowledge of the authors, no public benchmarking has been performed using the windAI_bench dataset with contemporary models. As such, this paper aims to provide a benchmark for windAI_bench to verify its applicability within the field.

In order to validate the methodology applied to benchmarking windAI_bench, the second dataset being investigated is the AirfRANS dataset published by Bonnet et al. [2]. AirfRANS contains 1,000 aerofoils and is benchmarked for use in Machine Learning (ML) by using four separate architectures. Importantly, whilst it includes training and testing loops, AirfRANS does not incorporate validation with unseen shapes and outside the aerofoils shape distribution. This means that the ability of any of the benchmark models to generalise to completely unseen data has not been ver-

ified. To keep the results in line with those for similar datasets, the models used will be the same as those in Bonnet et al. [2]. One point worth noting is that whilst these datasets are designed for wind turbines, they are also applicable to the design and evaluation of tidal turbines. To account for scalability issues, Reynolds number calculations would thus need to consider flow characteristics and operational velocities at which tidal turbines usually operate.

As mentioned, the computational requirement of generating CFD data means that the data is regularly unattainable in large enough quantities for data-driven NNs to be viable. This paper aims to investigate the amount of data required to attain reasonable results. Therefore, the performance of each model has been investigated over 5, 20, 55 and 150 aerofoils. These numbers were chosen to be roughly equal to an exponential scale as $5\approx e^{1.5}$, $20\approx e^3$, $55\approx e^4$ and $150\approx e^5$. It is important to validate the results from this against a sample set of aerofoils used neither in the training nor testing of the model, i.e., a validation dataset, to ensure that the model is not over-fitting to the data presented.

II. RELATED LITERATURE

Artificial Neural Networks (ANNs), or simply NNs, are universal function approximators [10] used within ML for the prediction of specified functions. NNs are called so because they are based on the design of the biological neural networks found within a brain, with neurons and synapses represented by the neurons that make up each layer of the NN and the connections that connect them, a diagram of which can be seen in Figure 1. The network is divided up into three categories of layers (input, hidden and output) each of a user specified width, i.e. how many neurons in each layer. Each connection contains a weight (w) which can be within the range $0 \le w \le 1$. These are multiplied by the output from the neuron it comes from (x) before being summed up with the remaining connections along with a bias factor (b), as seen in Eq. 1.

$$y = b + \sum_{i=1}^{n} w_i x_i \tag{1}$$

where y is the value of the new node, n is the number of nodes in the previous layer.

These weights and neuron values are used to affect the final output of the NN and this output is then compared to a 'ground truth' value, i.e. the 'real' value usually taken from the data. Then, after finding the gradient of the 'Loss', or difference between these two values, the weights and biases of the network are updated using an optimisation algorithm such as ADAM [11]. The data loaded into the model is split into training and testing data: the former is used to optimise the weights and biases of the model: whilst the latter is used to check the ability of the model to generalise to unseen data. It is worth noting that incorporating a testing split of data is not sufficient to claim the model can generalise to any unseen data.

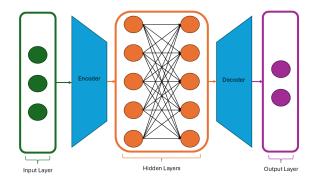


Figure 1: A Fully Connected Neural Network with an encoder and decoder

It is used to ensure that the model does not 'overfit' to the training data. This is then repeated until a point defined by the user, usually either when a certain number of epochs, or iterations, have been performed or when the loss has gone below a pre-determined threshold.

Some NNs make use of an encoder and a decoder within the code. The encoder block takes the input data and abstracts it into a *latent* vector, similar to the first two layers in Figure 1, whereas the decoder takes the latent vector and aims to reconstruct the input as the output. In this paper, there is an encoder before each model and a decoder after it. They are both defined using MLP and are trained along with the model.

There are a variety of ways in which NNs have been implemented in the disciplines surrounding renewable energies and fluid flow. In [12], NNs are used to model the flow of fluid within a combustor. The model uses experimental data from Particle Image Velocimetry (PIV) in order to predict the behaviour of said fluid. An older paper, [13], published in 2001, looks at a variety of applications for NNs within the renewable energy industry, but with a focus on solar power. The model performed notably well across all experiments, with a loss of between 1-9% accuracy.

One variation on the typical format of the NN is the Graph Neural Network (GNN) [14], which differs slightly from a regular NN as the data is defined with nodes connected to each other via edges, similarly to the layout of a mesh. As such, to determine the values of each node, the information is propagated to and from the neighbouring nodes iteratively. [15]

GNNs have been increasingly used for the prediction of fluid dynamics such as in Li et al. [16], and Peng et al. [17]. Ogoke et al. [18] use Graph Convolutional Networks (GCNs), first introduced by Kipf et al. [19], to predict the drag coefficient (C_D) , from a similar input to both windAI_bench and AirfRANS. The main difference in methodology that Ogoke et al. use is they are predicting solely C_D , and not the entire fluid behaviour.

The dataset being used to compare the benchmarks of windAI_bench [1] to is AirfRANS, published by Extrality [2], due to its pre-existing benchmark of fluid flow over an aerofoil. It includes 1,000 aerofoils that have been benchmarked using four architectures: a

basic MLP, PointNet [20], GraphSAGE [21], and GUNet [22], which is a GNN-based variant of the UNet architecture [23]. As mentioned in the introduction, further insight is given into the two datasets in section III-A.

III. METHODOLOGY

The code for this paper was done using PyTorch and PyTorch Geometric [24] and will be made publicly available after acceptance.

A. Data

WindAI_bench includes two different models, transition and turbulent, both run by a 2D Hamiltonian Strand Navier-Stokes solver (HAM2D). Specifically being used is the airfoil_2k subsection of the dataset, which contains 1830 unique aerofoil shapes at 25 angles of attack, ranging from -4° to 20° , for 3 separate Reynolds numbers. This gives a total of 45,750 simulations to train with, making the dataset comparable to that used by [25]. In order to give an insight into the amount of data required for reasonable results to be achieved by the models, only a limited number of aerofoils (across every angle of attack (AoA)) was utilised within this paper. Four sets of simulations was performed with 5, 20, 55 and 150 aerofoils for each of the models, the number of foils for each set has been chosen in line with an exponential scale, such that: $5 \approx e^{1.5}$, $20 \approx e^3$, $55 \stackrel{?}{\approx} e^4$ and $150 \approx e^5$. Whilst a full breakdown of the data can be found on windAI_bench's GitHub [1], for the purposes of this paper, the data used was the $Re = 3e^6$ data generated by the turbulence model. This includes, for each AoA, the C_L and C_D values, the 'landmarks' array which is a coordinate file for each point along the aerofoil, and the physical parameters at each node: namely the x and y positional co-ordinates, the density of the fluid, the x and y components of the momentum, the total energy, and the vorticity.

The airfoil_2k data was stored as a single H5 data structure, from which the data relevant to this paper was processed and stored into separate H5 files, one for each aerofoil. H5 files make use of HDF5, a 'Hierarchical Data Format' that is able to store large amounts of data that can be easily read and indexed [26].

The dataset was also normalised, making it nondimensional as this reduces any potential inaccuracies caused by performing mathematical operations on largely differing values. The normalised values are found as in Eq. 2.

$$\rho^* = \frac{\rho}{\rho_\infty}; \quad \sigma^* = \rho^* \frac{U}{a_\infty}; \quad \omega^* = \frac{\omega}{a_\infty}$$
 (2)

where, ρ is density, σ , momentum, U, velocity magnitude, ω , vorticity, a, speed of sound, and ∞ denotes the free-stream conditions, which in this case means $\rho_{\infty}=1.225kgm^{-3}$ and $a_{\infty}=340.15ms^{-1}$. As such, the momentum is given with respect to its relative Mach Number (Ma), meaning the magnitude of the free-stream momentum is equal to 0.1Ma within the simulations used for training.

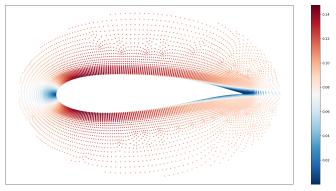


Figure 2: Point cloud of aerofoil #18 with reduced nodes for training, showing the ground truth for the Magnitude Momentum at an AoA of 0°

As the interactions between the fluid and the foil determine the aerodynamic properties of the aerofoil, the most important points in the domain are those at the surface of the aerofoil itself. The geometry information was not present within the mesh itself, but was defined as an array of 2D Cartesian co-ordinates in a separate geometry file labelled 'landmarks'. As such, determining which points lay on the surface was done by using the Python package, Shapely. Polygon. [27] A polygon object is generated using the 'landmarks' data, resulting in a shape made of straight lines going from point to point instead of the smooth curve expected of a foil. As such, none of the mesh points lie on the polygon itself and a buffer zone is implemented in order to check which points are closest to the foil. Whilst this inclusion of the buffer zone works with a high accuracy, it is not without drawbacks. Most often happening at the leading edge of the aerofoils, there are often a couple of extra nodes 'caught' in the buffer zone. However, after testing a range of buffer distances, $3.5e^{-6}m$ was found to be the best compromise between losing nodes at the upper and lower surfaces and catching extra nodes at the leading edge. Introducing this separation between surface, volume, and total areas was also implemented in the AirfRANS paper, and as such, it allows for a comparison of the effect of training on these areas and further comparison of the two benchmarks.

The errors on the nodes farther away from the aerofoil were typically very low when compared to those close to the aerofoil, causing the overall error of the model to be artificially low, creating a large error at the points in which this paper is interested. Therefore, when loading the data, all points outside a radius of 0.7m, centred at the midpoint of the chord length (c=1m), were omitted, noticeably increasing the error of the system, but ensuring better results at the points of interest. A graphical interpretation of this can be seen in Figure 2.

B. Models

In order to ensure that the results are comparable to those presented in other datasets, models have been taken from the AirfRANS dataset benchmark paper [2]. One of the most notable differences between the two datasets is the properties used to define the fluid flow. Airfrans has the models outputs/targets as velocity, pressure, and viscosity at each node, while windAI_bench uses density, momentum, energy, and vorticity. The inputs for this model are then based on the requirements of the outputs and for each node containing: positional co-ordinates, free-stream density, inflow momentum components, the initial energy and vorticity of the system set to ε (= $1e^{-10}$), and the negative signed distance function which determines the distance from the node to the aerofoil. [28] The data was then normalised again using just the mean and standard deviation of the training dataset to reduce errors in the training.

The loss criterion used for the models during training is the Mean Squared Error (MSE) as outlined in Eq. 3.

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2$$
 (3)

where, Y is the ground truth value from the dataset and \hat{Y} is the predicted value.

C. Post Processing

To properly assess the effectiveness of the trained models, a validation check is performed using 60 unseen aerofoils to assess the ability of the models to generalise to relevant unseen data. The metrics of interest for this analysis are the root mean squared error (RMSE) of both the prediction and the C_L calculated from the predicted flow. RMSE is more favourable than MSE as it converts the results back to the original scale and units of the target value, making it an easier metric to interpret. The formulation for the RMSE can be seen in Eq. 4.

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2}$$
 (4)

Using the predicted values of vorticity at the aerofoil, C_L is calculated using the panel method [29] and then compared to the values provided by the windAI_bench dataset via the RMSE.

The panel method is a method that allows for the approximation of forces acting upon an object in a flow by segmenting that object, in this case an aerofoil, into discrete surface elements, panels, and prescribing a flow element to it [29]. Using the volumetric vorticity (ω) predicted at each node on the aerofoil, the total circulation (Γ) for the aerofoil can be calculated by integrating ω over the total area defined by the aerofoil as seen in Eq. 5.

$$\Gamma = \iint_{A} \omega . dA \tag{5}$$

This allows for the calculation of the Lift (L) and coefficient of lift C_L acting on the aerofoil to be calculated as seen in Eqs. 6 & 7 [29]:

$$L = \rho U_{\infty} \Gamma \tag{6}$$

$$C_L = \frac{L}{\frac{1}{2}\rho U_{\infty}^2 c} = \frac{2\Gamma}{U_{\infty}c} \tag{7}$$

where U_{∞} is the free stream velocity, ρ is the fluid density and c is the chord length, which in this case is 1.

IV. RESULTS

As mentioned earlier in Section III-B, MSE was used to calculate the loss during the training and testing cycle, as is common practice, after which the final losses were converted into RMSE and presented in the tables below, in order to provide a more insightful view into accuracy. The graphs below are laid out as follows: for each total number of aerofoils, the first table contains the RMSE during testing at points on the aerofoil surface (Surface) and across all other points that are not touching the aerofoil (Fluid). Then, the last three columns of the first table contain the results from the validation as laid out in Section III-C, with the RMSE at the surface and in the fluid (the same points highlighted in the test RMSE) as well as the RMSE of the prediction of C_L averaged over all 60 aerofoils (1,500 simulations). The second and third sections for each table contain a breakdown of the RMSE during testing of each of the variables being predicted, namely density (ρ) , x-momentum (σ_u) , ymomentum (σ_v), energy (e), and vorticity (ω).

Table I shows the results for the tests trained on 5 aerofoils, Table II for those trained on 20, Table III for those trained on 55, and Table IV for those trained on 150.

It is worth noting that each number of aerofoils as displayed is not the same as the amount of data points the models are trained on as each foil has data for 25 AoAs. Therefore, the number of simulations each is trained on is: 125 for Table I, 500 for Table II, 1375 for Table III, 3750 for Table IV.

For all results, a lower score is better, and the lowest for each section has been highlighted in bold. All simulations were run on Windows 11 Home with PyTorch version 2.5.1, utilising Intel(R) Core(TM) i9-10980XE CPU @ 3.00GHz, NVIDIA GeForce RTX 3080 GPU, and 120GB of RAM.

V. DISCUSSION

There are a few key results to note from the outputs of the models, as shown in Tables I-IV. Firstly, despite their relative complexity and consistently high performance across the training regime, both the GraphSAGE and GUNet models achieve relatively high errors in the validation training set, regularly over double that of PointNet and MLP. One likely reason for such behaviour is that both GraphSAGE and GUNet are overfitting to the data in training; this means the models are memorising the data they are being trained on, instead of learning the patterns or trends within the data.

Table I shows very little correlation across the results, with all models scoring the lowest RMSE for at least one metric. This clear discrepancy when compared to the rest of the Tables indicates that 5 Aerofoils is insufficient to reliably predict anything.

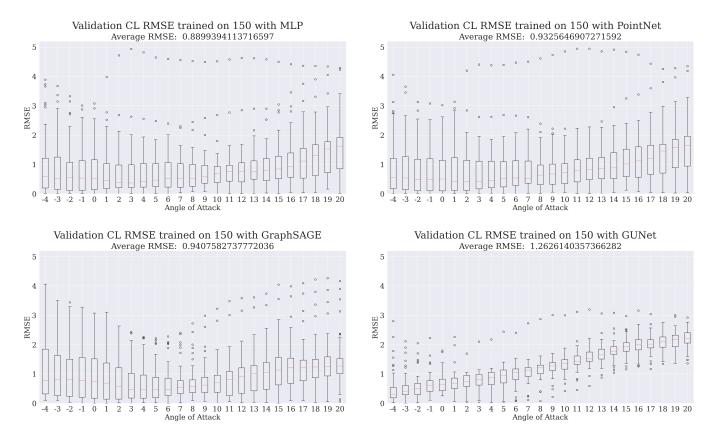


Figure 3: Box Plots showing the RMSE between the calculated and real C_L for all 60 validation aerofoils at each angle of attack trained on 150 aerofoils

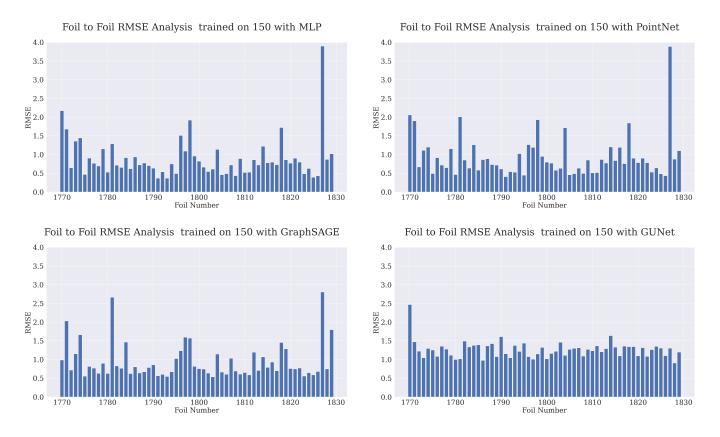


Figure 4: Bar Chart showing the RMSE between the calculated and real C_L for all 60 validation aerofoils

Table I: Table of RMSEs produced from models training and testing on 5 aerofoils

5 Aerofoils					
Model	Test RMSE		Validation RN		ЛSE
	Surface	Fluid	Surface	Fluid	CL
MLP	0.847	0.388	1.254	0.467	1.162
PointNet	1.302	0.484	1.643	0.603	0.953
GraphSAGE	0.750	0.390	2.299	2.920	1.219
GUNet	0.702	0.361	2.207	0.913	1.250
Validation RMSE Per Variable at Surface					
Model	ρ	σ_u	σ_v	e	ω
MLP	0.686	0.541	0.426	0.771	2.514
PointNet	0.644	1.117	0.906	0.748	3.233
GraphSAGE	1.399	0.742	0.753	1.618	4.554
GUNet	1.019	0.950	0.664	1.183	4.536
Validation RMSE Per Variable in Fluid					
Model	ρ	σ_u	σ_v	e	ω
MLP	0.432	0.489	0.375	0.431	0.581
PointNet	0.418	0.564	0.497	0.434	0.943
GraphSAGE	1.571	1.030	0.738	1.867	5.921
GUNet	0.627	0.859	0.654	0.653	1.478

Table II: Table of RMSEs produced from models training and testing on 20 aerofoils

20 Aerofoils					
Model	Test RMSE		Validation RM		ISE
	Surface	Fluid	Surface	Fluid	CL
MLP	0.422	0.234	0.640	0.263	0.933
PointNet	0.491	0.255	0.727	0.315	0.811
GraphSAGE	0.348	0.201	2.095	0.843	0.982
GUNet	0.422	0.247	1.917	0.949	1.290
Validation RMSE Per Variable at Surface					
Model	ρ	σ_u	σ_v	e	ω
MLP	0.432	0.150	0.097	0.477	1.266
PointNet	0.442	0.403	0.271	0.485	1.405
GraphSAGE	1.324	0.785	0.430	1.488	4.145
GUNet	1.247	0.532	0.369	1.405	3.799
Validation RMSE Per Variable in Fluid					
Model	ρ	σ_u	σ_v	e	ω
MLP	0.272	0.285	0.243	0.272	0.241
PointNet	0.284	0.335	0.286	0.288	0.373
GraphSAGE	0.745	0.675	0.703	0.746	1.222
ĜUNet	0.673	0.621	0.683	0.687	1.652

Table III: Table of RMSEs produced from models training and testing on 55 aerofoils

55 Aerofoils						
Model	Test RMSE \		Valid	lidation RMSE		
	Surface	Fluid	Surface	Fluid	CL	
MLP	0.341	0.140	0.456	0.181	0.894	
PointNet	0.351	0.148	0.431	0.178	0.817	
GraphSAGE	0.248	0.109	1.934	0.778	2.045	
GUNet	0.279	0.130	1.921	0.648	1.733	
Validation RMSE Per Variable at Surface						
Model	ρ	σ_u	σ_v	e	ω	
MLP	0.314	0.108	0.078	0.348	0.897	
PointNet	0.261	0.139	0.109	0.288	0.863	
GraphSAGE	1.266	0.522	0.326	1.413	3.836	
GUNet	1.148	0.495	0.240	1.287	3.894	
Validation RMSE Per Variable in Fluid						
Model	ρ	σ_u	σ_v	e	ω	
MLP	0.190	0.198	0.159	0.190	0.165	
PointNet	0.163	0.212	0.157	0.167	0.184	
GraphSAGE	0.734	0.694	0.692	0.707	1.013	
GUNet	0.553	0.571	0.556	0.547	0.927	

Table IV: Table of RMSEs produced from models training and testing on 150 aerofoils

150 Aerofoils						
Model	Test RMSE		Validation RMSE		ЛSE	
	Surface	Fluid	Surface	Fluid	CL	
MLP	0.204	0.024	0.355	0.134	0.890	
PointNet	0.309	0.033	0.400	0.147	0.933	
GraphSAGE	0.154	0.019	1.833	0.743	0.941	
GUNet	0.215	0.026	2.251	0.510	1.264	
Validat	Validation RMSE Per Variable at Surface					
Model	ρ	σ_u	σ_v	e	ω	
MLP	0.235	0.084	0.055	0.259	0.705	
PointNet	0.266	0.109	0.076	0.296	0.790	
GraphSAGE	1.263	0.489	0.269	1.406	3.594	
GUNet	0.741	0.636	1.222	0.920	4.694	
Validation RMSE Per Variable in Fluid						
Model	ρ	σ_u	σ_v	e	ω	
MLP	0.141	0.152	0.109	0.141	0.126	
PointNet	0.153	0.160	0.124	0.152	0.143	
GraphSAGE	0.749	0.650	0.623	0.745	0.914	
GUNet	0.347	0.336	0.404	0.349	0.885	

Table V: The inference time (time taken per epoch) for each model

Inference Time (ms)				
MLP	PointNet	GraphSAGE	GUNet	
1.90	4.31	3.14	44.16	

Across the overall validation metrics, MLP predicts the values at the surface and the fluid the most accurately six out of eight times, in Tables I, II, and IV. Interestingly, in Table II despite being slightly less accurate at the surface and the fluid, PointNet outputs a more accurate C_L . Tables III and IV, however, show the expected results, with the model most accurately able to predict the vorticity at the surface, resulting in the most accurate C_L .

As evidenced in Figure 9, and Tables I-IV, it is only MLP that becomes consistently more accurate as the number of aerofoils increases. Interestingly, the point of lowest RMSE, across all experiments, is for PointNet using 20 aerofoils, despite the RMSE for ω being higher than that of MLP in Table II. Notably, in Table II, MLP has the lowest RMSE for every metric, aside for C_L , a pattern shared, with the overall RMSEs of Table I. The fact that this pattern disappears as the number of aerofoils is increased could indicate that there is still a lack of sufficient training data required to provide consistent results. Further investigation is required to verify the cause of these inconsistencies.

GraphSAGE and GUNet both experience large increases to the validation RMSE during the 55 aerofoil experiments, despite seeing a consistent decrease in test RMSE from experiment to experiment. These results combined with the longer inference required due to the more complicated nature of the models (as shown in Table V) indicate a lack of suitability of GUNet and GraphSAGE for this problem statement. Whilst they train well, all evidence points to the models over-fitting to the training data.

The foil-to-foil comparison, shown in Figure 4, provided further insight into the effect that the physical properties of each aerofoil had on the predictions of the

models. The geometries of the most and least accurate four foils can be seen in Figures 5 and 6, respectively. When the validation aerofoils were compared to each other, four foils clearly stood out as having the highest errors across the models, whilst there were a large number of foils that resulted in relatively low RMSE.

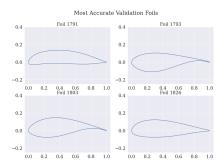


Figure 5: The four aerofoils with the lowest RMSEs across the models when trained on 150 aerofoils

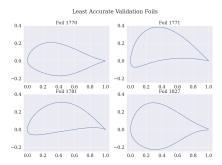


Figure 6: The four aerofoils with the highest RMSEs across the models when trained on 150 aerofoils

The differences between the two set of aerofoils are clear to see, the foils that performed better are thinner and have less of a camber. When comparing the RMSEs at each AoA for the less accurate foils, as shown in Figure 7, there is a visible correlation between the shape of the foils and whether the model was more accurate at low or high AoAs. The foils that were more rounded, with a larger lower camber were more accurate at lower angles of attack, whilst the foils with a larger mean camber were more accurate for higher angles of attack. Whereas the blades with a lower average RMSE for C_L in Figure 8 are all less thick. Interestingly, the better performing blades all have points of lowest RMSE between 10° and 15° before having their highest RMSE at 20° .

In Zhang et al. [30] found that a 'drastically varying surface curvature in the leading edge' [30] caused early flow detachments along the surface of the aerofoil. This separation causes the flow to reverse direction [30] which is more difficult for the models to predict than a simple and cohesive flow behaviour.

The relative thickness of aerofoil also plays a role in the accuracy of the model. Zhang et al. [31] found that, at low Re, a larger relative thickness causes a larger laminar separation bubble and affects the location of flow separation. Both of these factors effect the overall performance of the foil, as well as the models ability to predict the required physical parameters. Further to this, the aerofoils themselves have been defined in this dataset as a set of discrete points, which will introduce errors into the predictions. Whilst detrimental to the output of the results when compared to real world testing, this limitation is one that is shared with CFD [32] and as such should not affect the results listed in Section IV.

One minor issue found when analysing the aerofoil from the dataset is that none of the data points include the name of the aerofoil, making it difficult to verify the split of foils the model is being trained on. The model could, for example, be getting trained on a disproportionate number of NACA foils which would cause a lesser quality of generalisation to unseen, non-NACA aerofoils.

A limitation of this paper was the selection of the aerofoils used in training; to allow the models to be compared, they were all trained on the same data, i.e., the first *nth* foils in the dataset. It is usually good practice to randomise the dataset to reduce errors from training on too similar data. This could be further reason for the inconsistencies found in Tables I and II.

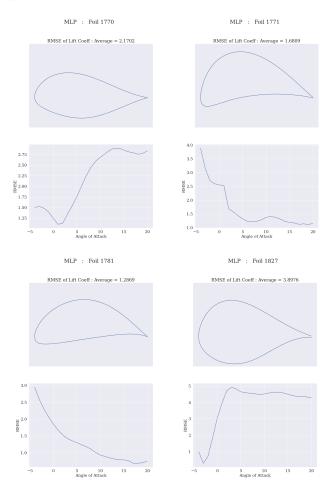


Figure 7: The four least accurate aerofoils shape and RMSE for C_L across all four models acquired using MLP trained on 150 foils

Unsurprisingly, the predictions generated by the models are somewhat compromised by the implementation of more extreme aerofoil designs. If NNs are

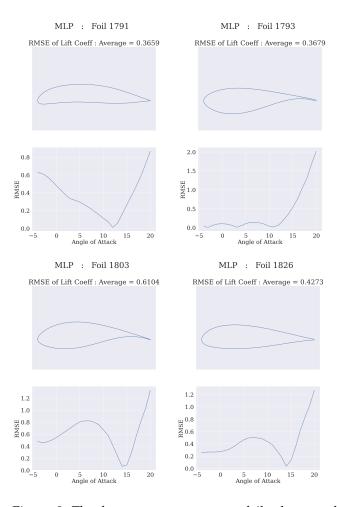


Figure 8: The four most accurate aerofoils shape and RMSE for C_L across all four models acquired using MLP trained on 150 foils

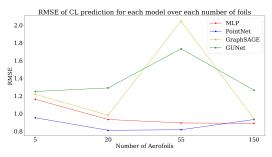


Figure 9: Comparison of the RMSE of C_L predicted by each model against how many aerofoils the models were trained upon

to be a viable tool in the design process of aerofoils, they need to be able to generalise to any foil they are given robustly. Common practice is simply to use more data, but, as highlighted earlier, this is not an effective strategy given the lack of data available and the intense computational load that NNs require.

As stated earlier in Section II, Bonnet et al. [2] provided a benchmark to data similar to that used for this paper, with the main differences being the variables calculated. Bonnet et al. also used the models to predict C_L and C_D directly, whereas this paper calculated them using the outputs. Whilst the results are comparable for the fluid predictions, the largest

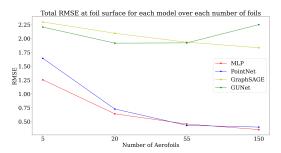


Figure 10: Comparison of the RMSE at the surface of the aerofoils predicted by each model against how many aerofoils the models were trained upon

difference in accuracy lies in the predictions of the coefficients. One of the biggest drawbacks to using the panel method for calculating the coefficients is that it cannot be used to calculate the drag force acting on the aerofoil. Therefore, in order to use windAl_bench to find C_L and C_D , the coefficients must be predicted using the models. However, if less data is incorporated into the models' training, the predictions of C_L and C_D would be less accurate.

VI. CONCLUSION

This paper presents a benchmark for the windAI_bench dataset created by NREL [1]. It compares the predictions made by four models using a range of sample sizes to determine the models' efficacy on smaller data sets. The results from the models were then used to analytically calculate the lift coefficients of the aerofoils at a number of AoA.

This paper found that whilst GraphSAGE and GUNet performed well during the testing phase, they performed poorly during the validation, meaning their ability to generalise to unseen data was inadequate relative to MLP and PointNet. Whilst these results for the latter models were relatively good, MLP showed far more consistency across the separate experiments, with the total RMSE found at surface, fluid, and C_L decreasing with every increase of aerofoil number. To fully investigate the extent of this improving accuracy, research should be undertaken to verify the difference training on more than 150 aerofoils would make.

Whilst PointNet did show excellent results and would be a viable model to use with 55 foils (1375 simulations), other techniques for ensuring consistency would have to be investigated.

In consideration of future work, one possibility as to why the models are over-fitting is that they are being overtrained with too many epochs. While increasing the number of epochs will reliably decrease the training and testing losses, further study is required in order to provide further insight into its effect on the validation process.

REFERENCES

[1] Ramos, Dakota, Glaws, Andrew, King, Ryan, Lee, Bumseok, Doronina, Olga, Baeder, James, Vijayakumar, Ganesh, , Grey, and Zachary., "Airfoil computational fluid dynamics - 2k shapes, 25 aoa's, 3 re numbers," 02 2023. [Online]. Available: https://data.openei.org/submissions/5970

- [2] F. Bonnet, J. A. Mazari, P. Cinnella, and P. Gallinari, "AirfRANS: High fidelity computational fluid dynamics dataset for approximating reynolds-averaged navier-stokes solutions," 2022. [Online]. Available: https://arxiv.org/abs/2212.07
- D. Coles, A. Angeloudis, D. Greaves, G. Hastie, M. Lewis, MacKie, J. McNaughton, J. Miles, S. Neill, M. Piggott, D. Risch, B. Scott, C. Sparling, T. Stallard, P. Thies, S. Walker, D. White, R. Willden, and B. Williamson, "A review of the uk and british channel islands practical tidal stream energy resource," vol. 477. Royal Society Publishing, 2021.
- D. for Energy Security and N. Zero, "Contracts for difference (cfd) allocation round 6: results," 2024.
- [5] B. Mannion, S. B. Leen, and S. Nash, "Development and assessment of a blade element momentum theory model for high solidity vertical axis tidal turbines," Ocean Engineering, vol. 197, p. 106918, 2020. [Online]. Available: https://www. sciencedirect.com/science/article/pii/S0029801820300020
- [6] F. Mahmuddin, "Rotor blade performance analysis with blade element momentum theory," *Energy Procedia*, vol. 105, pp. 1123-1129, 2017. [Online]. Available: https://www. sciencedirect.com/science/article/pii/S1876610217305180
- [7] J. Morgado, R. Vizinho, M. Silvestre, and J. Páscoa, "Xfoil vs cfd performance predictions for high lift low reynolds number airfoils," Aerospace Science and Technology, vol. 52, pp. 207-214, 2016. [Online]. Available: https://www.sciencedirect.
- com/science/article/pii/S1270963816300839 [8] Y. Luo, Y. Chen, and Z. Zhang, "Cfdbench: A large-scale benchmark for machine learning methods in fluid dynamics,' 2024. [Online]. Available: https://arxiv.org/abs/2310.05963
- W. Xu, N. Grande Gutierrez, and C. McComb, "Megaflow2d: A parametric dataset for machine learning super-resolution in computational fluid dynamics simulations," in Proceedings of Cyber-Physical Systems and Internet of Things Week 2023, ser. CPS-IoT Week '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 100-104. [Online]. Available: https://doi.org/10.1145/3576914.3587552
- [10] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," Neural Networks, vol. 2, no. 5, pp. 359–366, 1989. [Online]. Available: https://www.sciencedirect.com/science/article/pii/ 0893608089900208
- [11] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017. [Online]. Available: https://arxiv.org/ abs/1412.6980
- [12] Z. Wang, K. Gong, W. Fan, C. Li, and W. Qian, "Prediction of swirling flow field in combustor based on deep learning," Acta Astronautica, vol. 201, pp. 302-316, 12 2022.
- S. A. Kalogirou, "Artificial neural networks in renewable energy systems applications: a review," pp. 373–401, 2001.
 [Online]. Available: www.elsevier.com/locate/rser
 [14] M. Gori, G. Monfardini, and F. Scarselli, "A new model for
- learning in graph domains," in Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005., vol. 2, 2005, pp. 729-734 vol. 2.
- [15] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE* Transactions on Neural Networks and Learning Systems, vol. 32, no. 1, pp. 4-24, 2021.

- [16] S. Li, M. Zhang, and M. D. Piggott, "End-to-end wind turbine wake modelling with deep graph representation learning,"
- Applied Energy, vol. 339, 6 2023.
 [17] J.-Z. Peng, Y.-Z. Wang, S. Chen, Z.-H. Chen, W.-T. Wu, and N. Aubry, "Grid adaptive reduced-order model of fluid flow based on graph convolutional neural network," *Physics of Communications* Fluids, vol. 34, no. 8, p. 087121, 08 2022. [Online]. Available: https://doi.org/10.1063/5.0100236
- [18] F. Ogoke, K. Meidani, A. Hashemi, and A. B. Farimani, "Graph convolutional networks applied to unstructured flow field data," Machine Learning: Science and Technology, vol. 2, 12 2021.
- [19] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 9 2016. [Online]. Available: http://arxiv.org/abs/1609.02907
- [20] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation, 2017. [Online]. Available: https://arxiv.org/abs/1612.00593
- W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," 2018. [Online].
- Available: https://arxiv.org/abs/1706.02216

 [22] H. Gao and S. Ji, "Graph u-nets," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri Salakhutdinov, Eds., vo 2019, pp. 2083–2092. vol. 97. PMLR, 2019, [Online]. Available: Jun pp. https://proceedings.mlr.press/v97/gao19a.html
- O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 5 2015. [Online]. Available: http://arxiv.org/abs/1505.04597
- M. Fey and J. E. Lenssen, "Fast graph representation learning with pytorch geometric," 3 2019. [Online]. Available: http://arxiv.org/abs/1903.02428
- M. A. Bouhlel, S. He, and J. R. Martins, "Scalable gradient-enhanced artificial neural networks for airfoil shape design in the subsonic and transonic regimes," Structural and Multidisciplinary Optimization, vol. 61, pp. 1363–1376, 4 2020.

 M. I. of Technology (MIT), "Introduction to hdf5," last
- Accessed on: 2025-04-15. [Online]. Available: https://web.mit. edu/fwtools_v3.1.0/www/H5.intro.html
- [27] S. Gillies, C. van der Wel, J. Van den Bossche, M. W. Taves, J. Arnott, B. C. Ward, and others, "Shapely," Apr. 2025. [Online]. Available: https://github.com/shapely/shapely
- S. Osher and R. Fedkiw, Constructing Signed Distance Functions. New York, NY: Springer New York, 2003, pp. 63–74. [Online]. Available: https://doi.org/10.1007/0-387-22746-6_7
 [29] J. Liburdy, "Intermediate fluid mechanics," Tech. Rep., 2021.
- W. Zhang, W. Cheng, W. Gao, A. Qamar, and R. Samtaney, "Geometrical effects on the airfoil flow separation and transition," Computers & Fluids, vol. 116, pp. 60-73, 2015. [Online]. Available: https://www.sciencedirect.com/science/ article/pii/S0045793015001292
- [31] D. Ma, Y. Zhao, Y. Qiao, and G. Li, "Effects of relative thickness on aerodynamic characteristics of airfoil at a low reynolds number," Chinese Journal of Aeronautics, vol. 28, no. 4, pp. 1003-1015, 2015. [Online]. Available: https://www. sciencedirect.com/science/article/pii/S1000936115001260
- P. R. Spalart and V. Venkatakrishnan, "On the role and challenges of cfd in the aerospace industry," *The Aeronautical Journal*, vol. 120, no. 1223, p. 209–232, 2016.