Aerospace Science and Technology 152 (2024) 109345

Contents lists available at ScienceDirect

Aerospace Science and Technology

journal homepage: www.elsevier.com/locate/aescte

A deep neural network reduced order model for unsteady aerodynamics of pitching airfoils

Giacomo Baldan *, Alberto Guardone

Department of Aerospace Science and Technology, Politecnico di Milano, Via La Masa 34, Milano, 20156, Italy

ARTICLE INFO

Communicated by Antonio Filippone

Keywords:
Dynamic stall
Pitching airfoil
Deep learning
Neural network
Reduced order model

ABSTRACT

A machine learning framework is developed to compute the aerodynamic forces and moment coefficients for a pitching NACA0012 airfoil incurring in light and deep dynamic stall. Four deep neural network frameworks of increasing complexity are investigated: two multilayer perceptrons and two convolutional neural networks. The convolutional framework, in addition to the standard mean squared error loss, features an improved loss function to compute the airfoil loads. In total, five models are investigated of increasingly complexity. The convolutional model, coupled with the loss function based on force and moment coefficients and embedding the attention mechanism, is found to robustly and efficiently predict pressure and skin friction distributions over the airfoil over the entire pitching cycle. Periodic conditions are implemented to grant the physical smoothness of the model output both in space and time. An analysis of the training dataset point distributions is performed to point out the effects of adopting low discrepancy sequences, such as Latin hypercube, Sobol', and Halton, compared to random and uniform sequences. The current model shows improved performances in predicting forces and pitching moment in a broad range of operating conditions.

1. Introduction

Dynamic stall can be encountered in a wide range of aeronautical applications, such as over blades of helicopter rotors and maneuvering fixed wing aircraft. It is of concern also for wind turbines and turbomachinery applications. Dynamic stall is a highly nonlinear unsteady aerodynamic phenomenon observed at large angles of attack, in combination with rapid variations of incidence. The ability to predict and mitigate dynamic stall results in an improvement of safety standards and also a better estimation of the flight envelopes [1]. The occurrence of dynamic stall is also one of the critical factor limiting the maximum speed of conventional helicopters.

Many experiments and computational simulations have been performed to investigate dynamic stall phenomena and study the flow around pitching and plunging airfoils [2]. Especially during the last decade, thanks to the advances in numerical methods and computational power provided by modern GPGPUs [3,4] together with optical measurements techniques [5], dynamic stall has received particular attention in the literature where considerable advances in modeling, prediction, and understanding are reported. Several aspects of dynamic stall have been numerically investigated ranging from aspect ratio [6] to sweep angle [7] and compressibility effects [8]. The current state-of-the-art numerical simulations consist in wall-resolved large eddy simulations (LES) of ramp-up motion for spanwise-extruded airfoils [9].

One key aspect, relevant to many practical applications, is the formulation of a reduced order model (ROM) for dynamic stall. Due to the large computational costs associated to dynamic stall CFD simulations, ROM are needed to reduce the computation cost of *e.g.* aeroelastic analysis or flight mechanics evaluations. Recent developments in dynamic stall ROM leverage on high-fidelity numerical simulations to better understand the underlying physics of dynamic stall and gather data from larger datasets. An example of a dynamic stall ROM, which uses Spectral Proper Orthogonal Decomposition (SPOD) to decompose the velocity flow field from a Delayed Detached Eddy Simulations (DDES) with SST model, has been developed in Refs. Avanzi et al. [10,11]. The approach by Avanzi et al. [10,11] defines an appropriate operating range of the filter's typical dimension to identify coherent structures in the process and reconstructs force and moment coefficients over the pitching cycle using a reduced set of modes.

A powerful tool that is becoming more and more popular in engineering applications is machine learning [12]. Recent developments in deep learning bring advanced and innovative approaches to im-

* Corresponding author.
E-mail address: giacomo.baldan@polimi.it (G. Baldan).

https://doi.org/10.1016/j.ast.2024.109345
Received 11 April 2024; Received in revised form 18 June 2024; Accepted 19 June 2024
Available online 26 June 2024

prove the efficiency, flexibility, and accuracy of the predictive models [13–16]. Some of the outstanding applications of deep neural networks (DNNs) in the domain of computational physics are solution of partial differential equations (PDEs), like Physics-Informed Neural Networks (PINNs) [17–20]. Other neural network architectures that allow to analyze information from large datasets are Convolutional Neural Networks (CNNs). They received increasing attention by the fluid-mechanics community for their ability in pattern recognition [21]. In addition, convolution layers are usually coupled with pooling and up-sampling layers, allowing to reduce or increase data size, respectively. The neural network capability of analyzing a large dataset has been of particular interest in dynamic stall phenomena since time-dependent series can be analyzed as a whole. Damiola et al. [22,23] proposed a State-Space Neural Network (SS-NN) model trained using sine sweep functions at several angle-of-attack ranges. The study shows that SS-NN can be a powerful tool to accurately predict the unsteady aerodynamic loads of a pitching airfoil, both in pre-stall and post-stall conditions. In particular, the model succeeds in correctly capturing highly non-linear flow features such as the delay of flow separation, and the formation and shedding of the dynamic stall vortex. A final example is presented by Eivazi et al. [24], where an autoencoder network is used for non-linear dimension reduction and feature extraction as an alternative to singular value decomposition (SVD). Then, the extracted features are used as an input for a long short-term memory (LSTM) network to predict the velocity field at future time instances. Solera-Rico et al. [25], Wang et al. [26] further confirmed the NN capabilities of predicting dynamic stall loads by comparison with non-intrusive reduced order models based on dynamic mode decomposition and proper orthogonal decomposition.

This work presents a deep neural network-based framework designed to predict force and moment coefficients throughout an entire pitching cycle, encompassing small oscillations and both light and deep dynamic stall conditions. The primary goal is to define a reduced order model that can efficiently predict the aerodynamic loads for integration into an aeroelastic code. Contrary to semi-empirical models that impose underlying a-priori laws, such as the Leishman-Beddoes one, and need to be tuned for each operating condition, the present approach does not assume any constraints. Four deep neural networks of increasing complexity are applied: two multilayer perceptrons and two convolutional neural networks. The operating conditions are defined by four parameters: the mean pitch angle α_0, the amplitude of the oscillation α_s, the reduced frequency k, and the free-stream velocity V_∞. In section 2, the dataset point sequences are reported with the numerical setup of computational fluid dynamics simulations to generate the datasets. In section 3, the four neural network frameworks are presented together with the enhanced loss function and the attention mechanism for the convolutional frameworks. After describing the numerical setup, the performances of the models are investigated in detail in the section 4, highlighting the pros and cons. Finally, section 5 reports the conclusions and the final remarks.

2. Aerodynamic dataset generation

The aerodynamic dataset for training the model is generated by solving the unsteady 2D RANS equations. The numerical setup is assessed against the experimental campaign presented in Lee and Gerontakos [27]. A detailed grid convergence analysis and a study of the effects of the turbulence model is reported by Baldan and Guardone [28]. The main settings of the reference test case are summarized in the following for completeness. The reference case is the NACA 0012 airfoil with a chord c of 0.15 m, undergoing a sinusoidal pitching motion, at reduced frequency k = 0.1 and Reynolds number Re = $1.35 \cdot 10^5$. The free-stream velocity is $V_\infty = 14$ m/s with a turbulent intensity equal to 0.08%, pressure is $P_\infty = 1$ atm, and the pitching frequency ω is set equal to 18.67 Hz. The mean angle of attach α_0 is 10°, while the angle oscillation amplitude α_s is 15°. An O-grid mesh has been used in the computations, with to 512 nodes over the profile and 128 nodes in the normal direction, named medium grid in Baldan and Guardone [29]. The element size at the leading edge is $2.0 \cdot 10^{-3}\, c$ while at the trailing edge is $5.0 \cdot 10^{-4}\, c$ according to the best practice. The $y^+ < 1$ requirement is always satisfied for the first cell layer at the wall, for all considered operating conditions. Numerical simulations are performed using ANSYS Fluent 2023R2. The unsteady incompressible RANS equations are solved using second-order upwind discretization and second-order implicit time integration scheme. Gradients are retrieved through a least square cell-based method, and fluxes are obtained with the Rhie-Chow momentum-based formulation. Pressure-velocity equations are solved using the SIMPLE method. A rigid motion of the entire grid is prescribed through a user-defined expression to allow the airfoil pitching, $\alpha(t) = \alpha_0 + \alpha_s \sin(\omega t)$. The SST model with the intermittency equation [30] is employed to close the RANS equations. Each cycle is discretized with 3 600 time steps. All the simulations are evolved until time convergence is reached between subsequent cycles, meaning that the obtained solution overlaps.

Table 1
Discrepancy for Latin hypercube, Sobol', and Halton sequences of the datasets in the unitary hypercube before scaling to real values.

Dataset size	Latin	Sobol'	Halton
30	$5.18 \cdot 10^{-3}$	$3.00 \cdot 10^{-3}$	$2.44 \cdot 10^{-3}$
40	$2.04 \cdot 10^{-3}$	$1.52 \cdot 10^{-3}$	$1.42 \cdot 10^{-3}$
50	$2.70 \cdot 10^{-3}$	$1.07 \cdot 10^{-3}$	$1.25 \cdot 10^{-3}$
100	$7.11 \cdot 10^{-4}$	$3.36 \cdot 10^{-4}$	$2.67 \cdot 10^{-4}$
216	$2.52 \cdot 10^{-4}$	$8.49 \cdot 10^{-5}$	$8.02 \cdot 10^{-5}$

The design space is limited to three variables to reduce the dimensionality of the problem and decrease the number of performed simulations in all the analysis related to the random sequences and the preliminary study on the neural network architectures. Thus, the input parameters are the free-stream velocity V_∞ that can vary between 12 and 22 m/s, the pitching amplitude angle α_s that ranges from 5° to 15°, and the reduced frequency k extends from 0.1 to 0.2. All the other parameters are kept equal to the reference test case. The free-stream velocity allows to change the Reynolds number of the test case. The addition of this parameter aims at defining a more general model that can deal with compressible flows where the velocity is proportional to the Mach number having fixed the Reynolds over Mach ratio. When considering the full setup the additional input is the mean angle of the pitching cycle, α_0, and it varies in the range 0° to 10°.

To generate the training distribution, a uniform grid has been used to cover the input space utilizing 6 points for each variable, namely 216 points in total. Unfortunately, it is known that the uniform grid distribution performs poorly in terms of discrepancy especially for a limited number of points and large dimensionality [31]. To overcome this limitation, low-discrepancy sequences have been used. Low-discrepancy distributions, also called quasi-random sequences, are sequences of points strategically generated to achieve a more uniform coverage of the design space compared to purely random sequences. The objective is to minimize discrepancies or irregularities in the distribution of points, especially in higher dimensions. Unlike truly random sequences, low-discrepancy sequences systematically distribute points across different dimensions to create a more even sampling of the input space. This regular and deterministic pattern is advantageous in numerical methods and simulations where a more uniform sampling contributes to improved accuracy. In this work, Sobol', Latin hypercube, and Halton sequences [32] are used for training with different sizes: 30, 40, 50, 100 and 216. The discrepancy of each distribution is reported in Table 1. The model validation dataset is not obtained by splitting the training dataset to preserve the low-discrepancy property of the employed sequences. The model's performance is evaluated using two new sets of data, each containing 40 and 100 random points, respectively.

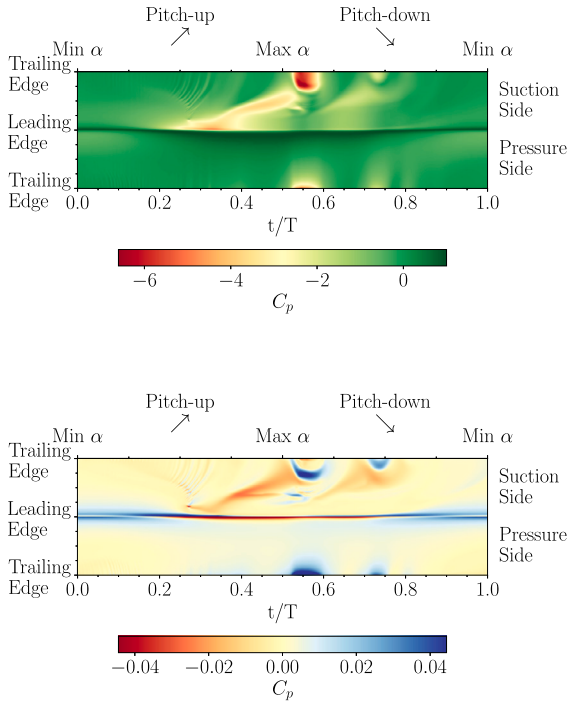


Fig. 1. Example of spatio-temporal contour of the post-processed C_p and C_f distributions over an entire pitching cycle.

When the full input space is considered, namely also the mean angle of the pitching cycle can be set, the 4-dimensional hypercube representing the operating conditions is covered using the Halton sequence with 64, 128, 256, and 512 points for the training dataset. The specific sequence has been selected among the others since it reaches the lowest errors as reported in result section 4. The testing dataset, instead, is built using a 64-point random sequence.

The outputs of each simulation, which populates the datasets, are the pressure coefficient and the skin friction coefficient distributions over the profile for each time step in one pitching cycle. The unstructured output from Fluent is mapped to the structured airfoil mesh using a k-d tree to search the nearest neighbors. This allows to organize the output in a 3-dimensional array and to compare results for different simulations. The first direction corresponds to the time step and ranges from 1 to 3 600. The second direction represents the airfoil surface coordinate. Indexes from 1 to 256 refer to data over the suction side of the profile while, from 257 to 512 to the pressure side. The size of the last direction is 2 and stores physical data, namely C_p and C_f. If the full dataset is considered, an additional dimension is present that identifies the simulation index. The post-processed value of the skin friction in each point is equal to the signed magnitude. The sign is set according to the direction computed from the x and y components of the C_f. The positive geometric direction is defined as the curvilinear coordinate starting from the leading edge towards the trailing edge for both sides of the airfoil. If the geometric direction and the C_f direction computed from the numerical simulations coincide, the sign is positive. Otherwise, the sign is negative. This allows to highlight the regions where the flow is detached and helps the neural networks to identify stalled portions. Finally, in 1, an example of spatio-temporal contour of the post-processed C_p and C_f distributions over an entire pitching cycle is available, highlighting the pitch-up and pitch-down phases and how the profile is mapped in the structured array.

3. Neural network frameworks

The term multilayer perceptron (MLP) is commonly used to identify a feed-forward artificial neural network, consisting of fully connected neurons with a non-linear activation function, made of at least three layers [33]. A convolutional neural network (CNN), instead, is a regularized type of feed-forward neural network which optimizes the filters, or kernels, through automated learning, whereas in traditional algorithms these filters are user-defined [34].

The model implementation and the training pipeline rely on the TensorFlow 2.11 deep-learning framework. Three different neural network frameworks are used in the first part of this work to study the learning capabilities of different architectures. The first two are MLPs while the third one is a CNN. In the second part, instead, the CNN framework is tested in the full 4-dimensional setup and an attention augmented convolutional neural network, AA-CNN, is proposed as a further improvement. The level of complexity increases moving from the first to the last one. The operating condition input in the assessment of the NN frameworks is made of three parameters, namely α_s, V_∞, and k, while in the last part α_0 is also included. All parameters are rescaled in the range $[0, 1]$ using their minimum and maximum values. The additional input parameter time t is present in the first framework. Time is scaled using the oscillation period, meaning that t can only assume values in the well-known range $[0, 1]$.

The output is different for all the first three frameworks, while it is the same for the CNN ones. In the first one, the values of C_l, C_d, and C_m are generated at each time level. Moving to the next framework, the same coefficients are computed but for the entire cycle leading to an output array of size $3\,600 \times 3$. Finally, in the CNN the output are the pressure and skin friction distributions over the airfoil for all time steps of size $3\,600 \times 512 \times 2$. Starting from the output distributions, the force and moment coefficients are straightforwardly retrieved from the airfoil geometry.

The simplest implementation, corresponding to the first framework, consists in an MLP neural network that predicts C_l, C_d, and C_m for a given operational condition and one temporal instant. For this reason, in the following is referred as *MLP Single*. It is characterized by three hidden layers of size 256, 1024, and 256 respectively. The activation function is Leaky ReLU (Rectified Linear Unit) which is commonly used to introduce non-linearity. The Leaky ReLU activation function allows a small, positive gradient when the input is negative, helping to address the vanishing gradient problem [35]. Mathematically, Leaky ReLU is defined as $f(x) = \max(\gamma x, x)$, where γ is set equal to 0.1.

Considering now the second MLP neural network, the structure is similar to the previous one since it always presents three hidden layers and adopts the same activation function. The differences concern the layer sizes that are: 256, 1 024, and 4 096. The last layer has size 10 800 that is reshaped to $3\,600 \times 3$ to cope with the output size. Due to the capability of predicting all the coefficients over one cycle is called *MLP All* for brevity.

Then, the Convolutional Neural Network (CNN) is considered. The first two hidden layers are formed by fully connected and Leaky ReLU-activated of size 256 and 184 320. After, it is reshaped to $45 \times 16 \times 256$. The first up-sampling layer increases the number of data points in the first direction by a factor of 5 while for the second direction they are doubled, leading to the first convolution layer with leaky ReLU activation. Four up-sampling and convolution layers follow, where the feature dimension is halved, and the other two directions are doubled until the output size is reached. The last convolution layer that reduces the 16 features to the pressure and friction distributions has a linear activation. In the actual implementation, before each convolution, an additional ad-hoc implemented layer is computed to enlarge the data and ensure periodicity. This step is very important from a physical point of view because it guarantees that there are no discontinuities over the airfoil and also that the start and the end of two cycles are coincident. In TensorFlow, the convolution layers are applied to the extended data with padding equal to *valid*. In a general setup, where the flow time history does not present any periodicity, the current approach can still be leveraged. The only difference lies in using a padding *same* for the time direction, namely the first and last values at the data boundaries

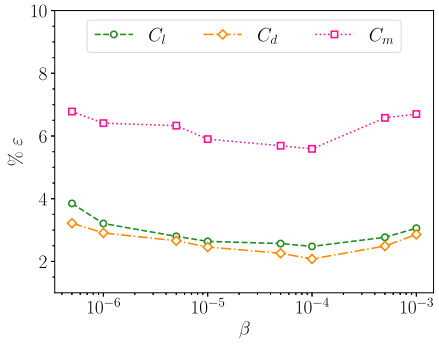


Fig. 2. Influence of β hyperparameter on the mean percentage error for C_l, C_d, and C_m. Training dataset is 216-point Halton and test dataset is 100-point Random.

are copied evenly. This is the first difference with respect to the other two frameworks that are not able to grant a physical meaning to the predicted data. Another advantage of having the C_p and the C_f distributions is the possibility of computing the moment coefficient with respect to an arbitrary point in the space that is not possible when directly learning the quarter-chord value. When the input includes the mean pitch angle, the actual input has size equal to 4 but all the other settings do not change.

Lastly, we present the AA-CNN in which the attention mechanism is added. It presents two differences with respect to the previous architecture. The first consists in the inclusion of three attention layers after the first convolution layers. The second, which is related to the memory requirements imposed by the attention computation, is the modification of the first up-sampling layer that only doubles the data points in the first direction. The fourth one, instead, now increases the first direction by a factor of five to complain with the output size. Going into the details of the attention layer, it consists of three 1x1 convolution layers to build the query, key, and value arrays used in the 16-head multi-head attention layer. After, the attention features are concatenated with the input and a final 1x1 convolution is performed to half the features.

The loss function $\mathcal{L}$ is the same in all the frameworks and it is the mean squared error (MSE).

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2, \tag{1}$$

where N is the number of data points, $\hat{y}_i$ is the predicted value for the i-th data point, and y_i is the ground truth value for the i-th data point.

For the CNN, an additional loss function has been implemented and tested trying to incorporate more physical information in the fourth model. Indeed, to build a robust model, only the data-driven approach is usually not sufficient and physics information have to be included in the model [36]. The objective is to include the mean squared force and moment coefficient error, MSE_{C_l,C_d,C_m}, in the loss function $\mathcal{L}$ in addition to the MSE presented before MSE_{C_p,C_f}.

$$\mathcal{L} = \text{MSE}_{C_p,C_f} + \beta \cdot \text{MSE}_{C_l,C_d,C_m} \tag{2}$$

The hyperparameter β controls the influence of the coefficient error in the total loss. The optimal value for β was found to be equal to $1 \cdot 10^{-4}$ in order to have a comparable error from the two contributions to the total loss function. A complete analysis on the influence of β is reported in Fig. 2. The shape of the airfoil and the mesh point positions are given as input to the neural network to compute the aerodynamic loads. The two models utilizing the CNN framework are distinguished based on their loss functions. The model employing the Mean Squared Error loss is referred as *CNN MSE*, while the one incorporating the improved loss function is designated as *CNN Coefficients*.

The AdamW algorithm [37] is employed, featuring a variable learning rate that starts at 10^{-4} and progressively decreases through a cosine decay to 10^{-11}. The training duration spans a maximum of 2 000 epochs. The complete training datasets are employed, while validation is performed on the random datasets. Training and inference wall-time for the analyzed frameworks is summarized in Table 2. The performances are evaluated using the 216-point Halton dataset for training and the 100-point random dataset for inference. A single NVIDIA A100 SXM4 40GB GPU is used in all the computations. Note that the improved loss function slightly increases the training time but does not influence the inference time still allowing a fast prediction. For the full setup, the wall-time is reported in Table 3. On the contrary of the previous consideration, the addition of the attention layers increases of around 20% the inference time. Finally, the training time is compared to the average time to compute a single CFD simulation using eight Intel Xeon Gold 6248 CPU @2.50GHz cores which corresponds to ≈ 2.2 hours. Focusing on the comparison between the computational time required to build the ROM, namely generate the dataset and training the NN, and the average CFD simulation, it can be noted that the ROM is convenient if the number of the evaluations is larger than the simulations used in the training dataset plus ≈ 5 equivalent simulations to train the NN. This number is in the order of one to few hundred.

Table 2
NVIDIA A100 training and inference wall-time. Performances are evaluated with the 216-point Halton dataset for training and the 100-point random dataset for inference. [CFD] is the equivalent number of CFD simulations.

Framework	Train time		Inference time
	[h]	[CFD]	[s]
MLP single	0.24	0.11	$2.45 \cdot 10^{-3}$
MLP all	0.32	0.15	$3.37 \cdot 10^{-3}$
CNN MSE	3.82	1.74	1.10
CNN Coefficients	4.01	1.82	1.10

Table 3
NVIDIA A100 training and inference wall-time. 4-dimensional input setup performances are evaluated with the 512-point Halton dataset for training and the 64-point random dataset for inference. Improved based loss is used. [CFD] is the equivalent number of CFD simulations.

Framework	Train time		Inference time
	[h]	[CFD]	[s]
CNN	9.11	4.14	1.11
AA-CNN	12.24	5.56	1.32

4. Results

This section delves into the performance of the presented neural network architectures in predicting the loads experienced by airfoils undergoing dynamic stall during pitching motion. It highlights both the strengths and the weaknesses of these networks comparing the results with high-fidelity numerical simulations. The analysis focuses on how the networks can handle unseen data and maintain accuracy across different operating conditions. Firstly, the results related to the effects of the quasi-random sequences are investigated using the different neural network frameworks. Finally, the analysis of the four-dimensional setup is presented using the best practices defined in the lower dimensional space.

The prediction error of an entire dataset is computed as the average error of the predicted coefficients per each simulation as reported in Equation (3). In detail, n_s is the number of simulations in the dataset and ${\varepsilon_s}^2$ is the squared error of the single snapshot per each coefficient. The error of the single simulation is derived as in Equation (4) in which

n_t is the number of time steps, namely 3 600, $\hat{x}_{s,t}$ is the predicted coefficient at time t in simulation s, and $x_{s,t}$ is the reference value. The reason for summing up the squared errors and dividing them by the sum of the reference values is linked to the occurrence of values close to zero. Indeed, when the ground truth value is close to zero, such as the C_m, a small discrepancy in the predicted value can lead to large relative errors giving a wrong estimation.

$$\bar{\varepsilon} = \sqrt{\frac{\sum_{s=1}^{n_s} {\varepsilon_s}^2}{n_s}} \tag{3}$$

$${\varepsilon_s}^2 = \frac{\sum_{t=1}^{n_t} \left(\hat{x}_{s,t} - x_{s,t}\right)^2}{\sum_{t=1}^{n_t} {x_{s,t}}^2} \tag{4}$$

The mean error provides overall performance of each framework. Since we are also interested in the performance for specific operating conditions, we investigated also some specific test cases of light and deep dynamic stall conditions.

4.1. Three-dimensional dataset and neural network framework analysis

Focusing on the preliminary study limited to the three-dimensional input, two dynamic stall test cases have been selected from the 100-point random dataset. The light dynamic stall is characterized by an oscillation amplitude of 8.62°, a 18.96 m/s free-stream velocity and a reduced frequency of 0.157. Deep dynamic stall, instead, occurs for an oscillation amplitude of 14.99°, a 15.71 m/s free-stream velocity and a reduced frequency of 0.105. In Fig. 3, the mean lift, drag, and moment coefficient relative errors are compared for all the analyzed frameworks with all the training and test datasets. The following two figures, 4 and 5, show the influence of the point distribution of the training datasets maintaining the number of points constant for the aerodynamic loads in the selected test cases. Figs. 6 and 7, instead, investigate the effect of the point number for the Halton sequence for the same test cases.

Focusing on the *MLP Single* neural network, the first aspect to notice is the sensitivity to the point distribution. Indeed, on the uniform grid the C_m error is not acceptable and it is several times larger than the other datasets. Moreover, for the same low discrepancy sequence, increasing the number of training points does not result in improving the accuracy, since the error is not following any particular trend. Considering now the predicted coefficients for the chosen test cases we can see that the trends are well captured for lift and drag. The pitching moment, instead, is in good agreement with the reference data only with the Halton and Latin hypercube datasets, even if for the deep dynamic stall case is completely missing the oscillations in both upstroke and downstroke phases. The framework suffers particularly in the C_m prediction. One of the reasons leading to large mean error is that the predicted curve is not a closed loop suggesting that the underlying physics is not learned. Another confirmation that the present architecture is not suitable to predict the unsteady loads is that increasing the number of neurons or layers does not improve the accuracy of the prediction.

The *MLP All* neural network outputs the force ad moment coefficients over an entire cycle. According to the mean errors reported in Fig. 3, the model better catches the physics of the problem. In particular, it is less sensitive to the point distribution used for the training and also the uniform grid distribution shows errors comparable to low discrepancy ones. Furthermore, enlarging the training datasets results in monotonically decreasing errors as expected. Despite the promising properties shown in the mean error map, when model predictions are analyzed, an additional issue is identified. Especially the drag and moment curves show spurious non-physical oscillations at low angle of attack making the model not suitable for the applications. Including additional layers in the framework and/or a larger number of neurons do not remove the oscillations. This behavior highlights that the NN architecture plays a crucial role in the prediction error and that the number of parameters is secondary.

The last neural network framework is based on convolution layers. The outputs of this framework are the C_p and C_f distributions. The required error convergence property increasing the number of points in each dataset and the low sensitivity to sequence type are satisfied as depicted in the error map. Looking closely in this regard, adding the loss based on the load coefficients further improves the performance. Even when the uniform grid distribution is adopted the error is comparable to the other distributions, confirming the necessity of include as many physical relations as possible in the model. The mean error in the *CNN Coefficient* framework is always lower than the *CNN MSE* one. Furthermore, the error always goes down when using a larger dataset with the more sophisticated loss. On the contrary, the *CNN MSE* has a peak in the mean error for 100-point Latin hypercube and 216-point Sobol' sequences. The Halton sequence is the one that reaches the lowest global error, but at the same time it is the one that has the largest error when using less points. The Latin and Sobol' ones, instead, present a more limited error variation while reaching satisfying performances for 100 and 216 points. Overall, the Latin hypercube grants better prediction quality with respect to Sobol. To further remark the effect of physics information in the model, in Figs. 4 and 5 it can be noted how the curves for all the three coefficients are closer to the actual solution and the differences of the datasets are less noticeable. A common issue is the ability to capture the instabilities near the trailing edge during the upstroke phase due to the transitioning of the boundary layer from laminar to turbulent. This behavior is mostly due to the up-sampling operations of the flow field that tends to smear out rapid oscillations. The other region where the error is larger coincides with the pressure peaks especially for the secondary vortex that involves highly non-linear phenomena and reaches C_p values close to the dynamic stall vortex that detaches first. A peculiarity that seems in contrast to what has been presented until now is the extremely low error for the uniform grid in the deep dynamic stall case. However, this behavior is due to another effect, that is the distance between the predicted point and the closest training point in the flight envelope. Specifically, for this case, the closest training point is located at $\alpha_s = 15°$, $V_\infty = 16$ m/s, and $k = 0.1$ which is particularly near to the deep dynamic stall test conditions.

4.2. Four-dimensional setup evaluation

Considering the complete input parameter space defined by four variables, a comparison is made between the coefficient errors of the standard *CNN* and *AA-CNN* models using the loss based on the load coefficients. Both architectures demonstrate the potential as robust alternatives for load prediction of an oscillating profile encompassing regimes from small oscillations to deep dynamic stall.

Fig. 8 reports the trend of the mean error $\bar{\varepsilon}$, as presented in the three-dimensional case, for the loads on the profile as a function of the training dataset size. The results demonstrate that the *AA-CNN* consistently achieves lower average errors compared to the standard *CNN*. This distinction is particularly evident for drag coefficient prediction, where the error reduction is in the $2-3\times$ range. Drag is inherently sensitive to both pressure and wall shear stress distributions. For lift and moment coefficients, the observed difference is less pronounced, remaining within the $20-50\%$ range. Notably, the C_m error exhibits a larger magnitude compared to the previously studied three-dimensional input scenario. This can be attributed to the inclusion of small oscillations at low-incidence angles, which generates large and high-frequency oscillations in the loads, which were not included in the prior operating condition set.

Fig. 9 offers a comprehensive investigation of this phenomenon visualizing the history of the loads across the entire pitching cycle. A crucial differentiating factor between the models lies in the presence or absence of spurious oscillations within the reduced-order model representation. The inclusion of the attention mechanism enables the *AA-CNN* to effectively extract the mean load trend. Con-

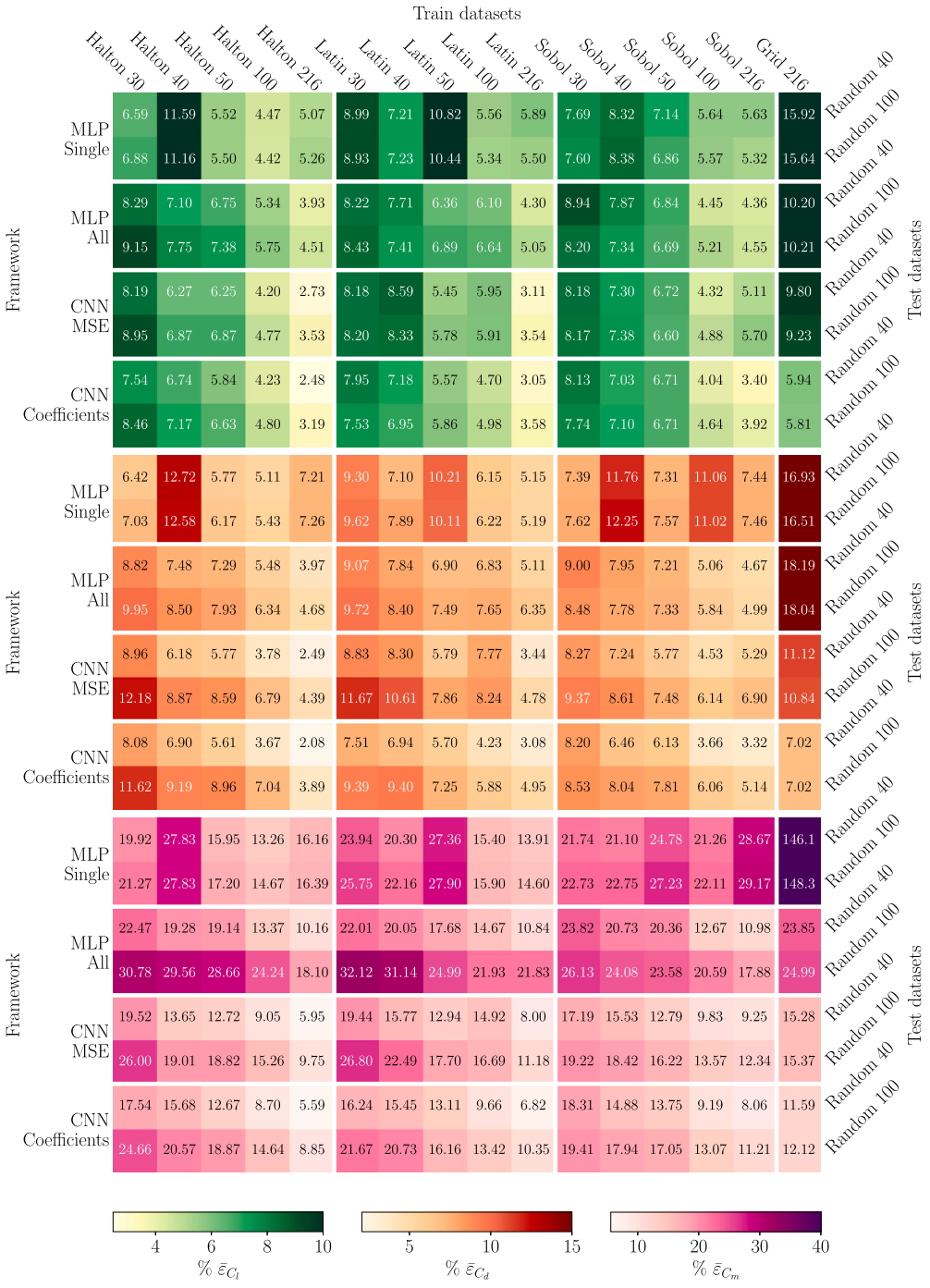


Fig. 3. Mean lift, drag, and moment coefficient error (Equations (3), (4)) comparison for the analyzed NN frameworks using different datasets for training and testing.

versely, the *CNN* retains the inherent oscillations in the data; however, their amplitude and frequency exhibit significant deviations from the reference solution. This discrepancy translates into a larger overall mean error.

Shifting focus to a light dynamic stall test case, the performance of both NNs is similar. Fig. 10 shows the loads as in the previous case. Notably, both architectures achieve excellent reconstruction capabilities, effectively replicating the high-fidelity simulation data. A closer examination of the results reveals that the attention-based NN exhibits a superior capability in reproducing the peak load values. Furthermore, a key strength of the *AA-CNN* lies in its ability to generate a near-identical solution using a training dataset comprised of only 64 data points. This highlights the model's data efficiency.

Finally, Fig. 11 depicts the deep dynamic stall conditions, where non-linear phenomena play a significant role. The reconstruction error remains substantial when the training dataset is limited to 64 points. However, a significant reduction is observed upon increasing the dataset size to 256 points. In this case, the advantage of the attention mechanism becomes more pronounced. The *AA-CNN* demonstrates a superior ability to capture the pressure peaks near the profile's nose, surpassing the performance of the standard *CNN* framework.

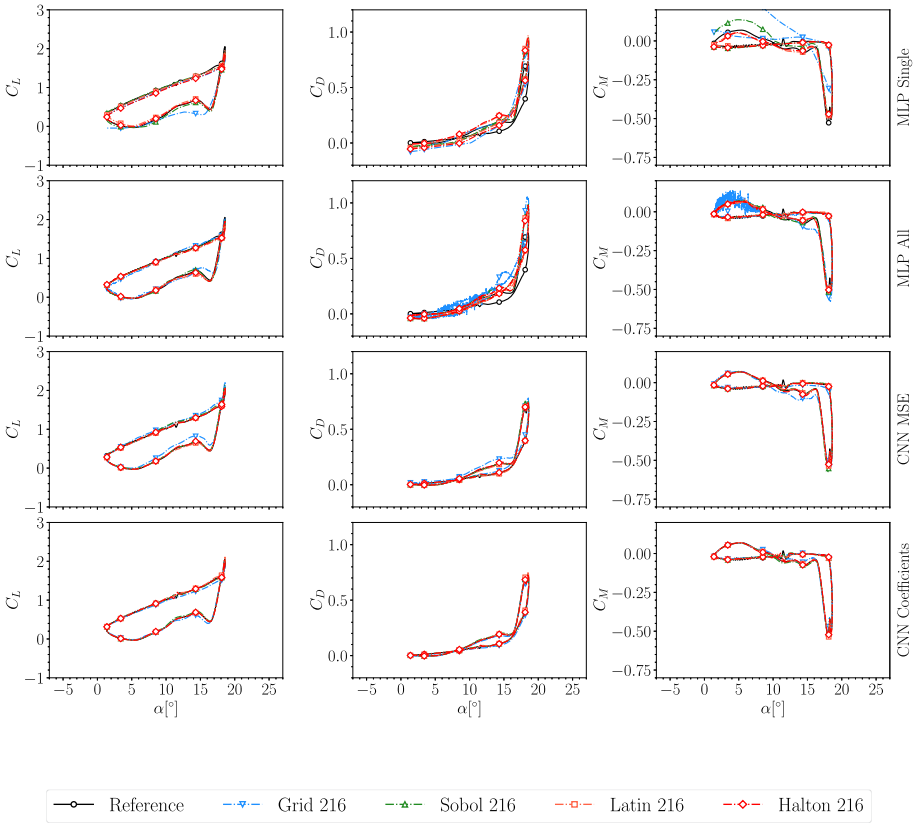


Fig. 4. Influence of quasi-random training sequence type on NN framework performances for a light dynamic stall example.

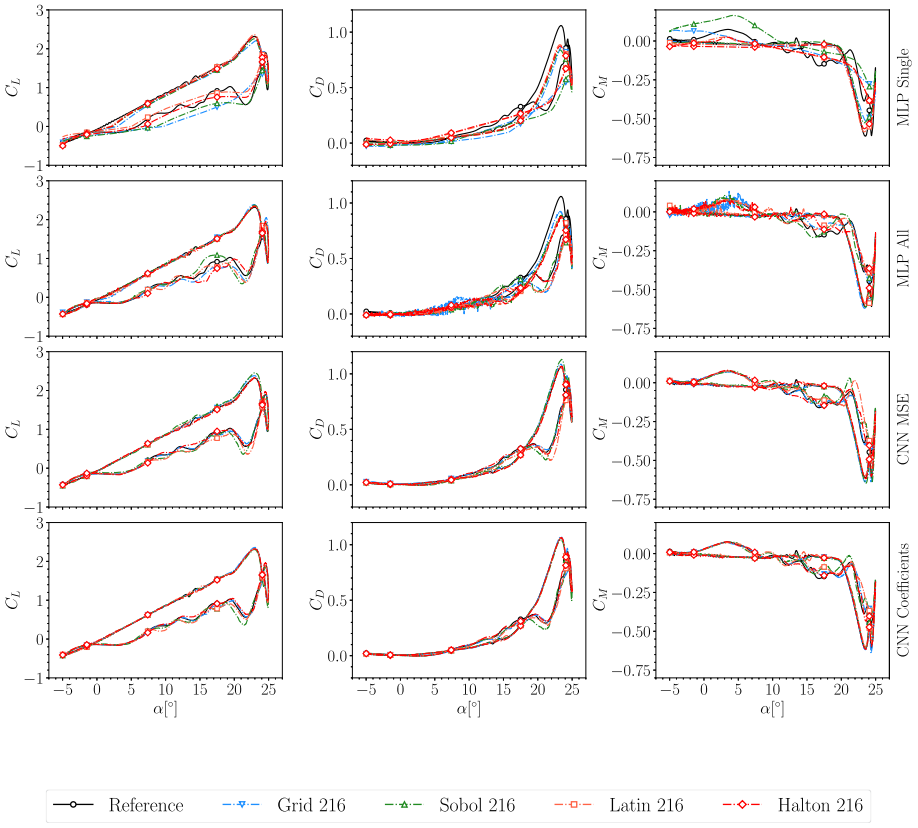


Fig. 5. Influence of quasi-random training sequence type on NN framework performance for a deep dynamic stall example.

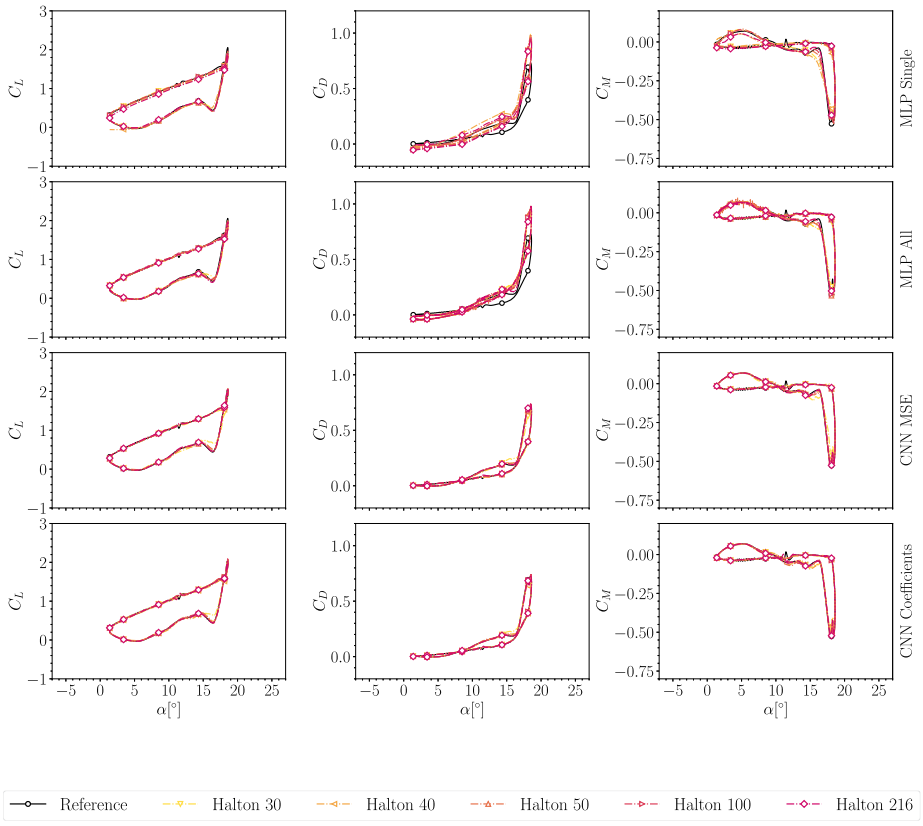


Fig. 6. Influence of Halton training sequence size on NN framework performances for a light dynamic stall example.

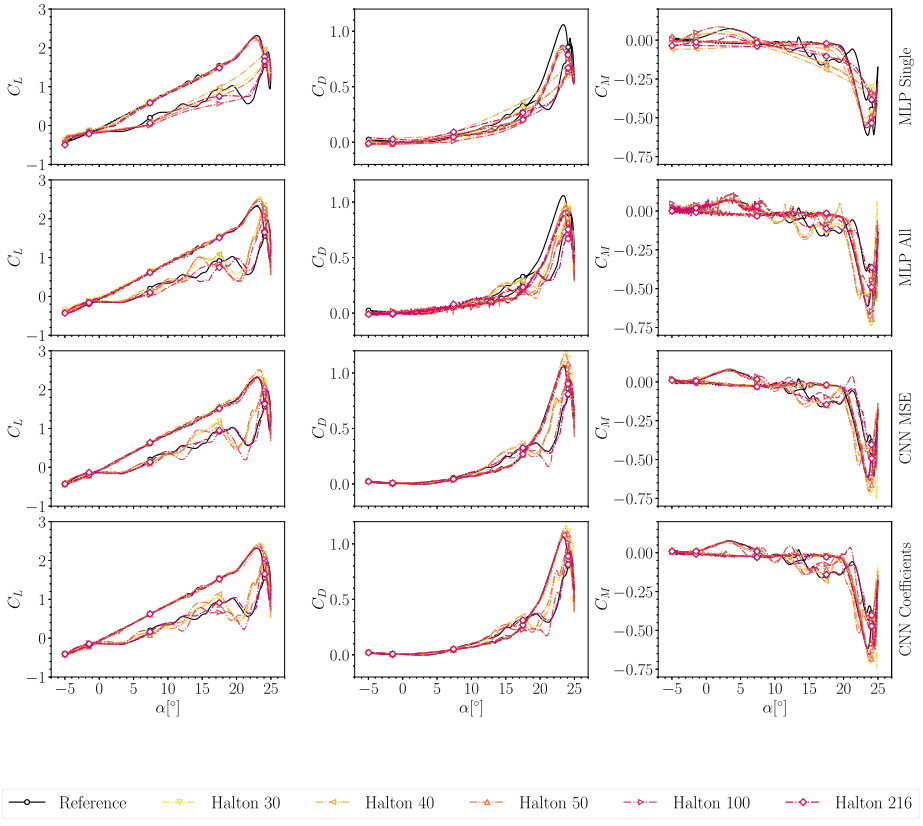


Fig. 7. Influence of Halton training sequence size on NN framework performances for a deep dynamic stall example.

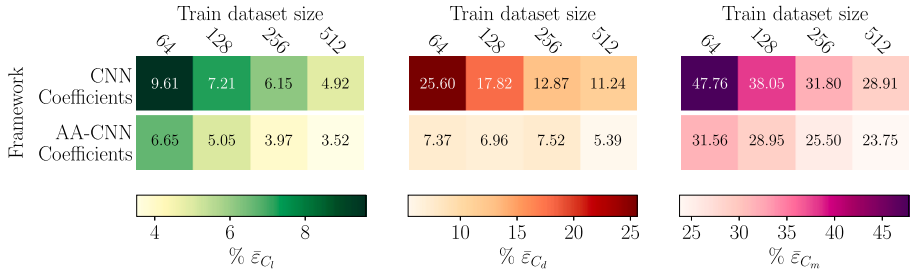


Fig. 8. Mean lift, drag, and moment coefficient error (Equations (3), (4)) comparison for the CNN and AA-CNN frameworks using different dataset sizes for training.

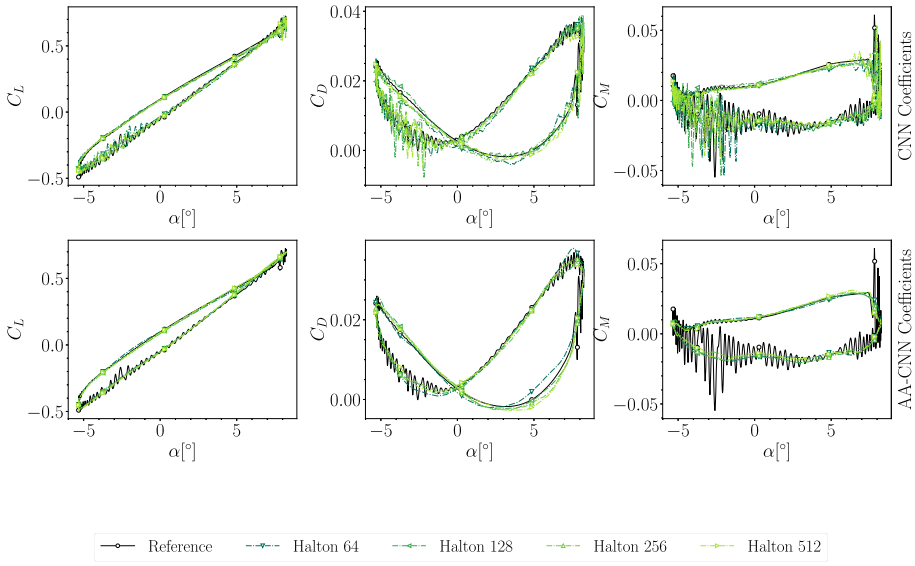


Fig. 9. Comparison of the force and moment coefficients for CNN and AA-CNN frameworks for a small oscillation test case.

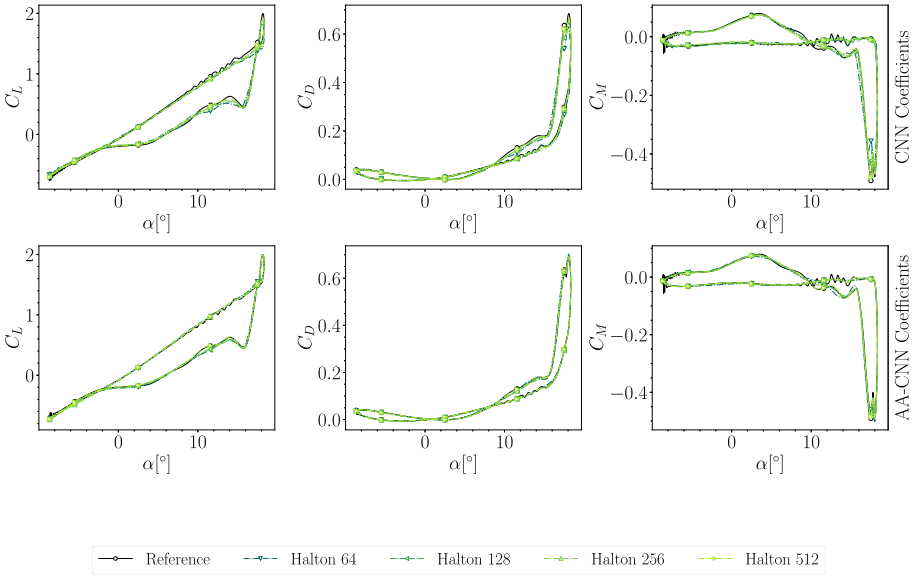


Fig. 10. Comparison of the force and moment coefficients for CNN and AA-CNN frameworks for a light dynamic stall test case.

5. Conclusions

This work presents a robust framework to fast predict aerodynamic loads for sinusoidal pitching airfoils incurring in light and deep dynamic stall. A data-driven reduced order model, based on deep neural networks and high-fidelity simulations, is build to accurately describe dynamic stall phenomena. Five different models, constructed upon four different deep learning frameworks, have been presented. A loss function that incorporates the computation of lift, drag, and moment coefficients in the network has been implemented, highlighting the advantages of including physical knowledge in the model. Moreover, a periodic condition has been implemented in the convolution layers to

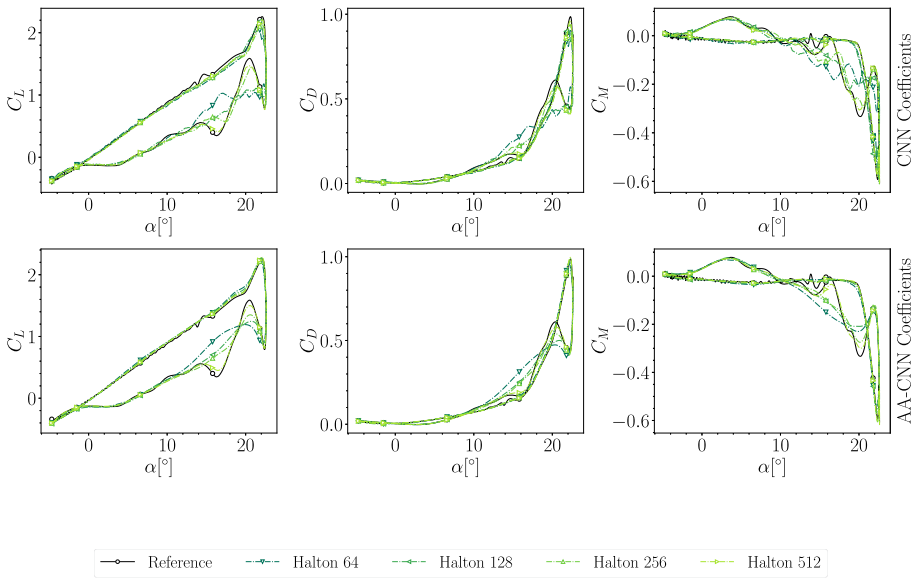
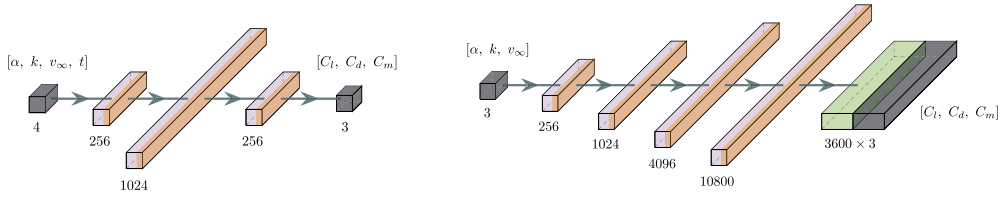
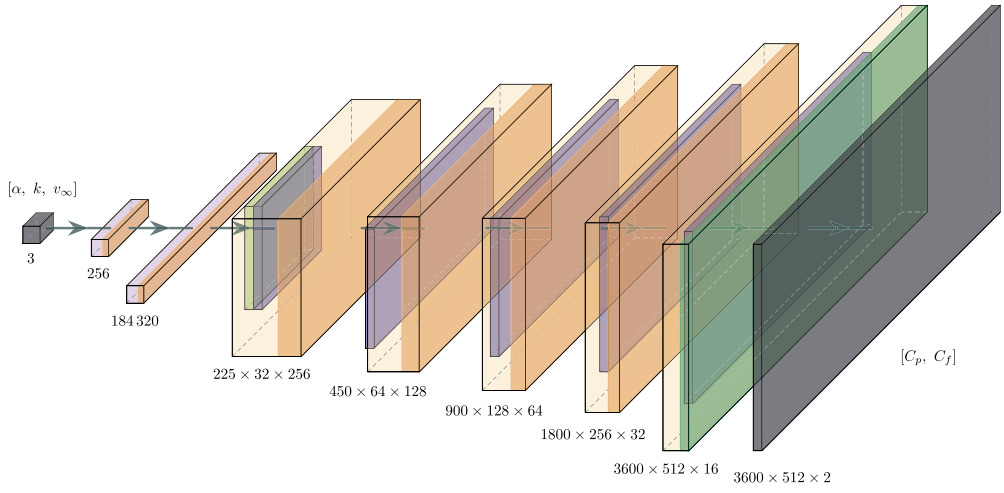


Fig. 11. Comparison of the force and moment coefficients for CNN and AA-CNN frameworks for a deep dynamic stall test case.



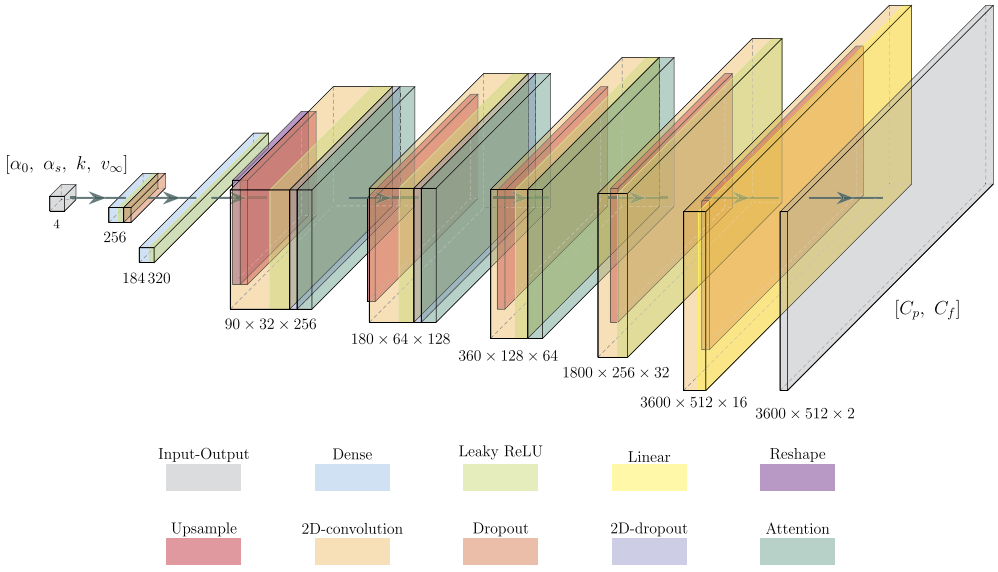
(a) MLP Single

(b) MLP All



(c) CNN

Fig. A.12. Neural network framework representations. Input parameters are: α_s, k, and V_∞. When present, t represents the time instant over the pitching period.



(a) AA-CNN

Fig. A.13. Neural network framework representation. Input parameters are: α_0, α_s, k, and V_∞.

ensure the physical meaning of the C_p and C_f outputs. Indeed, periodicity grants that there are no discontinuities in the distributions over the airfoil and also that the aerodynamic loads are continuous between subsequent cycles. Furthermore, an extensive analysis of the training dataset point distributions has been performed. A uniform grid distribution is compared against low discrepancy sequences, showing that the Halton sequence is the one that closely predicts the reference loads.

After the preliminary analysis using the three-dimensional operating condition space where the mean pitch angle is kept constant, we investigated the model in a possible real scenario in which all the input parameters are present. In this framework, the convolutional neural network, coupled with the improved loss function and possibly with the attention mechanism, showed improved performance in predicting aerodynamic loads for a broad range of operating conditions. Despite the use of incompressible CFD solutions, the properties of the presented framework are still valid for a full compressible setup, and the present models are candidates to fill the gap of next-generation reduced order models for dynamic stall phenomena.

CRediT authorship contribution statement

Giacomo Baldan: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Alberto Guardone:** Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgements

The authors acknowledge Leonardo SpA – Helicopter Division for granting access to the *davinci-1* supercomputer where all the computations are performed.

Appendix A. Neural network architectures

A representation is reported for the neural network architectures employed in this work. In detail, Figs. A.12a and A.12b represent the two MLPs while, the two CNN architectures are schematized in Fig. A.12c and A.13a.

References

[1] A.D. Gardner, A.R. Jones, K. Mulleners, J.W. Naughton, M.J. Smith, Review of rotating wing dynamic stall: experiments and flow control, Progress in Aerospace Sciences 137 (2023) 100887, https://doi.org/10.1016/j.paerosci.2023.100887.

[2] N.M. Khalifa, A. Rezaei, H.E. Taha, On computational simulations of dynamic stall and its three-dimensional nature, Phys. Fluids 35 (2023) 105143, https://doi.org/10.1063/5.0170251.

[3] F. De Vanna, G. Baldan, F. Picano, E. Benini, Effect of convective schemes in wall-resolved and wall-modeled LES of compressible wall turbulence, Comput. Fluids 250 (2023) 105710, https://doi.org/10.1016/j.compfluid.2022.105710, https://www.sciencedirect.com/science/article/pii/S0045793022003036.

[4] F. De Vanna, G. Baldan, F. Picano, E. Benini, On the coupling between wall-modeled LES and immersed boundary method towards applicative compressible flow simulations, Comput. Fluids 266 (2023) 106058, https://doi.org/10.1016/j.compfluid.2023.106058, https://www.sciencedirect.com/science/article/pii/S0045793023002839.

[5] L. Damiola, M.F. Siddiqui, M.C. Runacres, T. De Troyer, Influence of free-stream turbulence intensity on static and dynamic stall of a NACA 0018 aerofoil, J. Wind Eng. Ind. Aerodyn. 232 (2023) 105270, https://doi.org/10.1016/j.jweia.2022.105270, https://www.sciencedirect.com/science/article/pii/S016761052200366X.

[6] P.R. Hammer, D.J. Garmann, M.R. Visbal, Effect of aspect ratio on finite-wing dynamic stall, AIAA J. 60 (2022) 6581–6593, https://doi.org/10.2514/1.J062109.

[7] P.R. Hammer, D.J. Garmann, M.R. Visbal, Effect of aspect ratio on swept-wing dynamic stall, AIAA J. 61 (2023) 4367–4377, https://doi.org/10.2514/1.J063039.

[8] S.I. Benton, M.R. Visbal, Effects of compressibility on dynamic-stall onset using large-eddy simulation, AIAA J. 58 (2020) 1194–1205, https://doi.org/10.2514/1.J058681.

[9] R. Miotto, W. Wolf, D. Gaitonde, M. Visbal, Analysis of the onset and evolution of a dynamic stall vortex on a periodic plunging aerofoil, J. Fluid Mech. 938 (2022) A24, https://doi.org/10.1017/jfm.2022.165.

[10] F. Avanzi, F. De Vanna, Y. Ruan, E. Benini, Design-assisted of pitching aerofoils through enhanced identification of coherent flow structures, Designs 5 (2021), https://doi.org/10.3390/designs5010011, https://www.mdpi.com/2411-9660/5/1/11.

[11] F. Avanzi, F.D. Vanna, Y. Ruan, E. Benini, Enhanced identification of coherent structures in the flow evolution of a pitching wing, in: AIAA SCITECH 2022 Forum, 2022, https://arc.aiaa.org/doi/abs/10.2514/6.2022-0182.

[12] S.L. Brunton, B.R. Noack, P. Koumoutsakos, Machine learning for fluid mechanics, Annu. Rev. Fluid Mech. 52 (2020) 477–508, https://doi.org/10.1146/annurev-fluid-010719-060214.

[13] R. Vinuesa, S.L. Brunton, B.J. McKeon, The transformative potential of machine learning for experiments in fluid mechanics, Nat. Rev. Phys. 5 (2023) 536–545, https://doi.org/10.1038/s42254-023-00622-y.
[14] L.-W. Chen, N. Thuerey, Towards high-accuracy deep learning inference of compressible flows over aerofoils, Comput. Fluids 250 (2023) 105707, https://doi.org/10.1016/j.compfluid.2022.105707, https://www.sciencedirect.com/science/article/pii/S0045793022003000.
[15] L.-W. Chen, N. Thuerey, Deep learning-based predictive modelling of transonic flow over an aerofoil, arXiv:2403.17131, 2024.
[16] Q. Liu, N. Thuerey, Uncertainty-aware surrogate models for airfoil flow simulations with denoising diffusion probabilistic models, AIAA J. (2024) 1–22, https://doi.org/10.2514/1.J063440.
[17] H. Eivazi, M. Tahani, P. Schlatter, R. Vinuesa, Physics-informed neural networks for solving Reynolds-averaged Navier–Stokes equations, Phys. Fluids 34 (2022) 075117, https://doi.org/10.1063/5.0095270.
[18] M. Baldan, G. Baldan, B. Nacke, Solving 1D non-linear magneto quasi-static Maxwell's equations using neural networks, IET Sci. Meas. Technol. 15 (2021) 204–217, https://doi.org/10.1049/smt2.12022, https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/smt2.12022.
[19] M. Baldan, P.D. Barba, B. Nacke, Magnetic properties identification by using a bi-objective optimal multi-fidelity neural network, IEEE Trans. Magn. 57 (2021) 1–4, https://doi.org/10.1109/TMAG.2021.3068705.
[20] M. Baldan, P. Di Barba, D.A. Lowther, Physics-Informed neural networks for inverse electromagnetic problems, IEEE Trans. Magn. 59 (2023) 1–5, https://doi.org/10.1109/TMAG.2023.3247023.
[21] L. Guastoni, A. Güemes, A. Ianiro, S. Discetti, P. Schlatter, H. Azizpour, R. Vinuesa, Convolutional-network models to predict wall-bounded turbulence from wall quantities, J. Fluid Mech. 928 (2021) A27, https://doi.org/10.1017/jfm.2021.812.
[22] L. Damiola, J. Decuyper, M. Runacres, T.D. Troyer, Modeling airfoil dynamic stall using State-Space Neural Networks, in: AIAA SCITECH 2023 Forum, 2023, https://arc.aiaa.org/doi/abs/10.2514/6.2023-1945.
[23] L. Damiola, J. Decuyper, M.C. Runacres, T. De Troyer, Modelling the unsteady lift of a pitching NACA 0018 aerofoil using state-space neural networks, J. Fluid Mech. 983 (2024) A8, https://doi.org/10.1017/jfm.2024.148.
[24] H. Eivazi, H. Veisi, M.H. Naderi, V. Esfahanian, Deep neural networks for nonlinear model order reduction of unsteady flows, Phys. Fluids 32 (2020) 105104, https://doi.org/10.1063/5.0020526.
[25] A. Solera-Rico, C.S. Vila, M.A. Gómez, Y. Wang, A. Almashjary, S.T.M. Dawson, R. Vinuesa, β-variational autoencoders and transformers for reduced-order modelling of fluid flows, https://doi.org/10.48550/arXiv.2304.03571, arXiv:2304.03571, 2023.
[26] Y. Wang, A. Solera-Rico, C. Sanmiguel Vila, R. Vinuesa, Towards optimal β-variational autoencoders combined with transformers for reduced-order modelling of turbulent flows, Int. J. Heat Fluid Flow 105 (2024) 109254, https://doi.org/10.1016/j.ijheatfluidflow.2023.109254, https://www.sciencedirect.com/science/article/pii/S0142727X23001534.
[27] T. Lee, P. Gerontakos, Investigation of flow over an oscillating airfoil, J. Fluid Mech. 512 (2004) 313–341, https://doi.org/10.1017/S0022112004009851.
[28] G. Baldan, A. Guardone, Pattern recognition of the flow around a pitching NACA 0012 airfoil in dynamic stall conditions, in: Aeronautics and Astronautics, 2023.
[29] G. Baldan, A. Guardone, The effects of turbulence modeling on dynamic stall, https://doi.org/10.48550/arXiv.2404.14172, 2024.
[30] F.R. Menter, R.B. Langtry, S.R. Likki, Y.B. Suzen, P.G. Huang, S. Völker, A correlation-based transition model using local variables – Part I: model formulation, J. Turbomach. (2004) 413–422, https://doi.org/10.1115/1.2184352.
[31] D.G. Loyola R, M. Pedergnana, S. Gimeno García, Smart sampling and incremental function learning for very large high dimensional data, in: Special Issue on "Neural Network Learning in Big Data", Neural Netw. 78 (2016) 75–87, https://doi.org/10.1016/j.neunet.2015.09.001, https://www.sciencedirect.com/science/article/pii/S0893608015001768.
[32] L. Kocis, W.J. Whiten, Computational investigations of low-discrepancy sequences, ACM Trans. Math. Softw. 23 (1997) 266–294, https://doi.org/10.1145/264029.264064.
[33] G. Cybenko, Approximation by superpositions of a sigmoidal function, Math. Control Signals Syst. 2 (1989) 303–314, https://doi.org/10.1007/BF02551274.
[34] S. Albawi, T.A. Mohammed, S. Al-Zawi, Understanding of a convolutional neural network, in: 2017 International Conference on Engineering and Technology (ICET), 2017, pp. 1–6.
[35] B. Xu, N. Wang, T. Chen, M. Li, Empirical evaluation of rectified activations in convolutional network, https://doi.org/10.48550/arXiv.1505.00853, arXiv:1505.00853, 2015.
[36] M. Dias Ribeiro, M. Stradtner, P. Bekemeyer, Unsteady reduced order model with neural networks and flight-physics-based regularization for aerodynamic applications, Comput. Fluids 264 (2023) 105949, https://doi.org/10.1016/j.compfluid.2023.105949, https://www.sciencedirect.com/science/article/pii/S0045793023001743.
[37] I. Loshchilov, F. Hutter, Decoupled weight decay regularization, arXiv:1711.05101, 2019.