Journal of Applied Fluid Mechanics, Vol. 17, No. 1, pp. 60-74, 2024.
Available online at www.jafmonline.net, ISSN 1735-3572, EISSN 1735-3645.
https://doi.org/10.47176/jafm.17.1.2109

A Data-Driven Machine Learning Approach for Turbulent Flow Field Prediction Based on Direct Computational Fluid Dynamics Database

M. Nemati and A. Jahangirian†

Aerospace Engineering Department, Amirkabir University of Technology, Tehran, Iran

†*Corresponding Author Email: ajahan@aut.ac.ir*

ABSTRACT

A novel approach is presented for predicting compressible turbulent flow fields using a neural network-based data-driven method. Accurate prediction in turbulent regions heavily relies on the resolution of available data. Traditional methods, employing image-based techniques by mapping scattered computational fluid dynamics (CFD) data onto Cartesian grids, encounter data scarcity in critical areas such as the boundary layer and wake. Recently, convolutional neural networks (CNN) have gained prominence as the most widely referenced technique in fluid dynamics, utilizing flow field images as datasets for flow field prediction. However, CNN requires datasets with a high pixel density to enhance training accuracy in crucial regions, thereby increasing the input data volume and machine training time. To address this challenge, our proposed method deviates from using flow field images and instead generates datasets directly from the flow field properties of CFD grid points. By employing this approach, several advantages are realized. Firstly, the network benefits from the favorable characteristics of unstructured grids, such as varying point spacing near the object surface and in the far field, which effectively reduces the amount of input data and consequently the machine training cost. Secondly, the construction of the training dataset eliminates the need for interpolation or extrapolation, thereby preserving the accuracy of CFD data. In this case, a simple multilayer perceptron can be trained using the proposed dataset. Various flow field properties, including static pressure, turbulent kinetic energy, and velocity components, can be predicted with high accuracy within a few seconds.

Article History

Received June 26, 2023
Revised August 12, 2023
Accepted August 29, 2023
Available online November 1, 2023

Keywords:

Flow field prediction
Turbulent flow
Machine learning
Data-driven
Surrogate Models
Computational Fluid Dynamics

1. INTRODUCTION

Despite the availability of high-performance computing tools (Marshall et al., 1997; Moureau et al., 2011) and advanced computational fluid dynamics (CFD) techniques, optimization processes still face a significant computing time challenge due to the substantial number of numerical solution iterations required (Kashefi et al., 2021) . Machine learning has emerged as a valuable approach in fluid dynamics, particularly for addressing this issue by reducing computation time and enabling the rapid acquisition of flow field data (Hallock & Holzäpfel, 2018; Kavitha & Mukesh Kumar, 2018; Li et al., 2020).

One effective strategy involves combining Euler and RANS datasets (Ghoreyshi et al., 2013) or directly employing RANS flow field images through modern image processing techniques (Guo et al., 2016). These approaches serve the purpose of diminishing the time and complexity associated with constructing the model. The proliferation of diverse software tools has resulted in the accumulation of vast amounts of data generated from numerical modeling (Ansari et al., 2018; Yu et al., 2019). However, managing and analyzing such massive datasets can pose significant challenges, often leading to complexity and time-consuming processes.

In light of these challenges, researchers have recently turned their attention to the application of neural networks to expedite the determination of flow field parameters (Akbıyık & Yavuz, 2021; Brunton et al., 2020; Krizhevsky et al., 2017; Nagawkar & Leifsson, 2022; Thuerey et al., 2019; Wang et al., 2023; Wu et al., 2020a; Yuan et al., 2018). By leveraging neural networks, it becomes possible to obtain flow field predictions in

significantly shorter timeframes, offering a promising alternative to the computationally intensive numerical solutions.

One crucial aspect of employing neural networks for flow field prediction lies in the selection and utilization of suitable datasets. In image processing, the features of input images are initially extracted, after which the neural network is connected to the image through a Cartesian grid. This grid divides the image into smaller cells, with each segment connected to the neural network. However, due to the non-linear nature of the governing equations, determining the values of each region within the flow field becomes more intricate, complicating the integration of scattered CFD data with the convolutional neural network (CNN).

The utilization of the pixelation method has been observed in numerous studies to establish a connection between scattered CFD data and CNN (Hasegawa et al., 2020; Miyanawala & Jaiman, 2019). This approach involves projecting CFD data onto a Cartesian grid, enabling the application of conventional CNNs for data training. Guo et al. (2016) employed CNN to predict the velocity field of steady laminar flow over various bluff shapes. For training purposes, they utilized 100,000 velocity flow field images, each with a resolution of 256×128 pixels. In their study, the training process achieved an overall accuracy of approximately 98% for predicting the velocity flow field. In their image representation, the geometry was encoded using a signed distance function (SDF). However, due to the low resolution of the input images with isotropic pixels, the predicted flow exhibited a lack of precision in proximity to the body.

Bhatnagar et al. (2019) employed CNN to predict steady RANS flow fields of U, V, and pressure by training the network with flow field data encompassing 4 Reynolds numbers, 3 airfoil shapes, and 21 angles of attack. However, their work encountered challenges as the network training error was approximately (10^{-2}), leading to unacceptable absolute errors between the CFD and prediction results. Additionally, the limited number of samples within their dataset appeared to diminish the accuracy of the trained network.

Sekar et al. (2019) successfully reduced the training error by employing a larger dataset, leading to enhanced prediction quality. Their dataset consisted of 5280 elements, encompassing 6 angles of attack (AOA), 8 Reynolds numbers, and 110 NACA airfoil shapes. Each input flow field consisted of 216 × 216 pixels. The training focused on flow field properties such as P, U, and V velocities, resulting in the utilization of 180 million parameters in the network. The network training error reached approximately (10^{-6}). However, it is worth noting that the time required to train the model was approximately 62 days using an Intel Xeon 3.3 GHz processor, which is a considerable duration.

Li et al. (2020) employed CNN to train a model capable of predicting the flow field within an isolator using a dataset comprising various supersonic Mach numbers. Du et al. (2021) utilized a combination of MLP and recurrent neural network to predict pressure and velocity flow fields across a wide range of Reynolds and Mach numbers. (Jin et al., 2018) proposed a CNN-based model that leveraged pressure measurements around a cylinder to predict the velocity flow field. Wu et al. (2021b) developed a model utilizing a generative adversarial network (GAN) and CNN to predict the pressure flow field around several supercritical airfoils. (Kashefi et al., 2021) employed CFD data to train a deep network for laminar flow field conditions using the PointNet architecture (Qi et al., 2017).

The dataset utilized in their study consisted of various shapes, including circles, squares, triangles, rectangles, ellipses, pentagons, and hexagons. During the training process, each cross-section experiences a distinct laminar flow field condition. For predicting flow fields of unseen geometries such as airfoils, interpolation techniques were employed based on other flow field geometries. The dataset comprised a total of 2595 instances, with 2076 allocated for training and the remaining portion used for validation and testing purposes.

It is important to note that CNN, being a method primarily designed for image and pattern recognition (O'Shea & Nash, 2015), employs images (pixels) as its training dataset. Consequently, the crucial requirement for accurate flow field prediction in critical areas may not be adequately addressed due to the Cartesian representation of the input images. Additionally, using images as a dataset presents two significant challenges: training time, as demonstrated by Sekar et al. (2019), and prediction accuracy, as highlighted by Kashefi and Mukerji 2022.

In this article, we introduce the utilization of direct CFD data as a solution to overcome the aforementioned challenges. By solving the Navier-Stokes equations, flow field variables are obtained and stored within each cell of the computational grid. These variables, in addition to the Mach number and angle of attack, will serve as the dataset input for the neural network. The direct feeding of unaltered data into the neural network offers distinct advantages compared to image-based approaches. To construct the dataset, the compressible Navier-Stokes equations are solved using a finite volume cell-centered implicit scheme to acquire the external flow properties.

A two-equation $k-\varepsilon$ turbulence model is also used in conjunction with the wall function approach to model the Reynolds stress terms in the momentum equations. The neural network establishes a relationship between the airfoil geometry and input parameters, such as the Mach number and angle of attack, to predict the corresponding flow field properties. These flow field properties encompass pressure, velocity components, and kinetic energy. The dataset utilized in this study focuses exclusively on the RAE2822 airfoil geometry. Changes in the flow field are solely attributed to variations in the free-stream variables, namely the Mach number and angle of attack.

To validate the performance of the trained neural network, flow field predictions will be conducted for Mach numbers and angles of attack that were not included in the dataset. These predictions will be compared against

the corresponding results obtained from CFD simulations. These predictions will be compared against CFD results to evaluate the network's performance. The Mean Squared Error (MSE) will be used as a metric to quantify the error between the predicted flow field and the CFD data. Additionally, the absolute error distribution across the flow field will be analyzed to examine error propagation in various regions.

In the following section the general implementation road map of the proposed method is presented in section 2. Neural network architecture and hyper parameters settings are also discussed in section 3.

2. METHODOLOGY

2.1 The Concept of the Proposed Method

The proposed method aims to overcome the limitations of image-based machine learning approaches, as outlined below:

a) Interpolation or extrapolation involved in the pixelated images can introduce inaccuracies in the input data for machine learning. By utilizing CFD data instead, the need for interpolation and extrapolation is eliminated, resulting in more accurate predictions of CFD results. Moreover, employing the control volume method to discretize the governing flow equations during the data generation phase ensures the preservation of mass and momentum conservation. The dataset is generated from the values stored in these control volume cells, automatically preserving mass, momentum, and energy in the flow predicted by the trained model. In contrast to pixel-based methods, this data is free from artificial effects and can be directly utilized.

b) In image-based methods, every region within the flow field is uniformly discretized with the same dimensions, assuming equal importance throughout. However, in the CFD-based approach, the computational grid employs a finer density of points in critical areas and a coarser distribution in others. This data distribution allows for reduced computing time while simultaneously improving prediction accuracy, particularly in regions of significance such as the boundary layer.

By utilizing CFD data, the proposed method enables the use of a neural network with a simpler structure, leading to significantly reduced training time. This is due to the substantially smaller number of computational grid data points compared to the pixel-based image methods.

As previously mentioned, turbulent flows are characterized by high velocity gradients near the surfaces of objects, and accurately predicting these flows relies heavily on the data resolution within these regions. However, conventional image-based machine learning methods employ relatively coarse Cartesian cells throughout the domain, which do not provide the necessary resolution near the objects (Kashefi et al., 2021). Attempting to increase pixel resolution to capture finer details near objects results in a significant increase in the total number of pixels across the entire field. Thus, a trade-off is required between resolution in sensitive areas

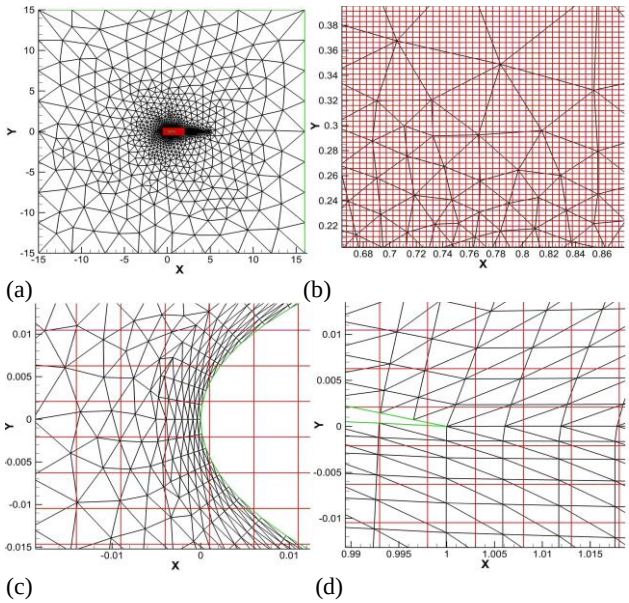


Fig. 1 Comparison between image-based (red) and CFD-based (black) methods. (a) full region, (b) red box outer boundary region, (c) leading edge and (d) trailing edge

and the total number of pixels to maintain the quality of the images.

Figure 1 illustrates a comparison between the CFD grid and the corresponding Cartesian cells utilized in the reference by Sekar et al. (2019), where high-quality images were employed during the training process. In Fig. 1(a), it is evident that the area covered by the image-based method is significantly smaller than the CFD solution area. However, the image-based method requires a larger number of inputs. The red box in Fig. 1(a), measuring 216×216, contains 46,656 pixels, whereas the total number of computational cells in the CFD method is only 5436.

Figures 1(b)-(d) demonstrate the resolutions of the data for the image-based method and the proposed method in different regions of the domain. The image-based method demonstrates a lack of adaptive data point distribution across the domain, resulting in cells located far from the object being of the same size as those near the object.

The flowchart for machine learning using the direct CFD dataset is presented in Fig. 2. Various values of Mach number and angle of attack are utilized to generate the training dataset. The Mach number is selected from the range of 0.2 to 0.5, while the angle of attack ranges from 1 to 5 degrees. The Latin hypercube sampling (LHS) method (McKay et al., 2000) is employed to ensure the selection of data from all defined ranges.

The values obtained for each flow parameter, such as pressure, kinetic energy, and velocities (u and v), are stored in numerical grid cells to serve as the input dataset for machine learning. The total number of computational cells in this particular flow field is significantly smaller, with only 5436 cells, in comparison to methods that utilize image-based datasets. This advantage is further enhanced by the fact that the training dataset is derived directly from the computational grid results, ensuring an appropriate data density in critical areas. This improved data distribution,

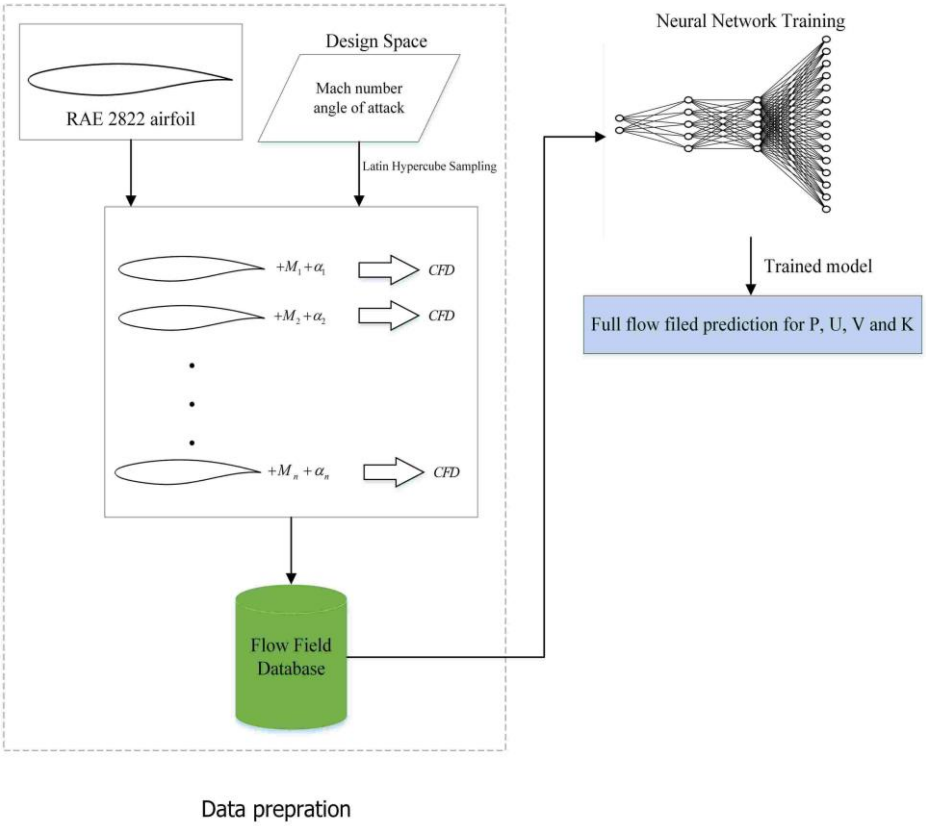


Fig. 2 Flowchart of machine learning based on direct CFD dataset

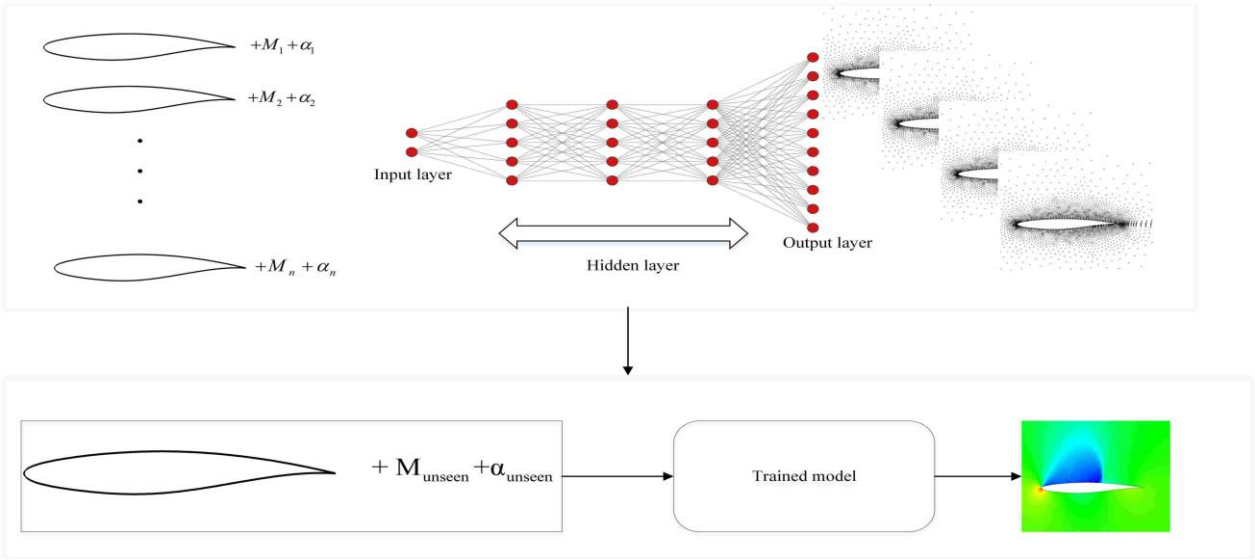


Fig. 3 Neural network's training and prediction

particularly for predicting turbulent flows, contributes to enhanced prediction accuracy.

Given that the input and output of the neural network are known, a supervised learning approach will be employed. As depicted in Fig. 3, the airfoil geometry, Mach number, and angle of attack will serve as the input, while the flow field, encompassing the values of flow parameters stored in the computational cells, will be the output of the machine learning process.

Upon completing the training process, the flow field can be predicted with high accuracy within a few seconds by providing new inputs that the machine has not encountered before. As demonstrated in subsequent sections, each flow parameter (p, k, u, v) may converge at different epoch values. Consequently, an independent machine, similar to the one depicted in Fig. 3, will be trained for each flow field.

2.2 Computational Fluid Dynamics Method and Dataset Preparation

The non-dimensional differential form of the compressible Reynolds- Averaged Navier–Stokes (RANS) equations in two dimensions can be expressed in the conservative form as:

$$\frac{\partial w}{\partial t}+\frac{\partial f^{I}}{\partial x}+\frac{\partial g^{I}}{\partial y}=\frac{M_{\infty}}{\mathrm{Re}_{\infty}}\left[\frac{\partial f^{v}}{\partial x}+\frac{\partial g^{v}}{\partial y}\right], \tag{1}$$

Where w is the vector of conserved variables, f^{I} and g^{I} are the inviscid fluxes defined as:

$$w=\begin{pmatrix}\rho\\ \rho u\\ \rho v\\ \rho E\end{pmatrix},\quad f^{I}=\begin{pmatrix}\rho u\\ \rho uu+P\\ \rho vu\\ \rho Eu+Pu\end{pmatrix},\quad g^{I}=\begin{pmatrix}\rho v\\ \rho uv\\ \rho vv+P\\ \rho Ev+Pv\end{pmatrix}. \tag{2}$$

In the above notation ρ, u, v, P and E are the non-dimensional density, velocity components, pressure and total energy. Viscous flux terms f^{v} and g^{v} are defined as:

$$f^{v}=\begin{pmatrix}0\\ \tau_{xx}\\ \tau_{xy}\\ u\tau_{xx}+v\tau_{xy}-q_{x}\end{pmatrix},\quad g^{v}=\begin{pmatrix}0\\ \tau_{xy}\\ \tau_{yy}\\ u\tau_{xy}+v\tau_{yy}-q_{y}\end{pmatrix}, \tag{3}$$

The Navier–Stokes equations are completed by the perfect gas equation of state:

$$P=\rho RT,\quad E=e+\frac{u^{2}+v^{2}}{2},\quad e=\frac{R}{\gamma-1}T \tag{4}$$

The no-slip boundary condition is employed at the surface boundaries, while non-reflecting boundary conditions based on characteristic analysis are applied in the far field. Additionally, wall function conditions are taken into account for near-wall turbulent calculations. A two-equation k-ε turbulence model is employed in conjunction with the governing flow equations (Jahangirian & Hadidoolabi, 2005). This method allows a Y^{+} between 30-50 that reduces the number of computational cells in the boundary layer.

Within the machine, each combination of the Mach number and angle of attack at the input corresponds to a specific flow field at the output. To simplify the training process, it is assumed that the airfoil geometry remains constant, allowing the Mach number and angle of attack variables to be the sole factors influencing the flow field. During the training process, it is important to define the range of variations for each parameter to determine the training domain.

In this study, a total of 25 Mach numbers ranging from 0.2 to 0.5 and 21 angle of attack values ranging from 1 to 5 degrees, with equal intervals, result in a dataset comprising 525 combinations of Mach numbers and angles of attack. For each pair of free-stream Mach number and angle of attack, the pressure distribution, turbulent kinetic energy, and the U and V velocity components in the flow field will be extracted from the CFD simulations conducted for the RAE2822 airfoil.

Figure 4 depicts the numerical grid employed to extract training data from the CFD simulations conducted around the RAE2822 airfoil. The dimensionless airfoil has a length equal to one, which is placed in a domain with

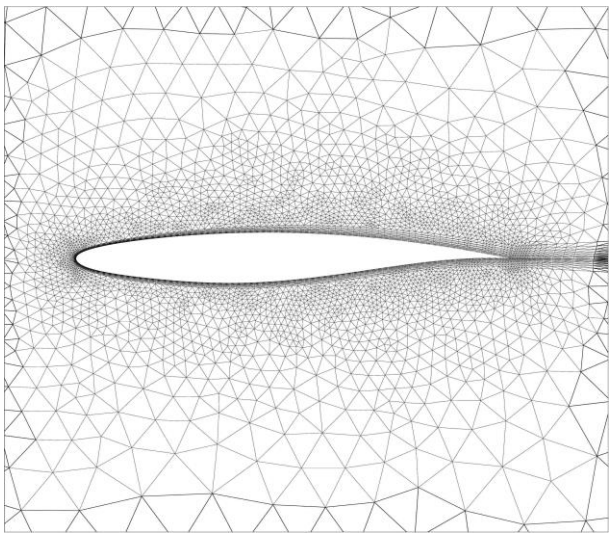


Fig. 4 Computational grid for RAE2822 used as neural network input

dimensions equal to 30 x 30. By utilizing an appropriate numerical grid, a greater amount of data is automatically provided in important areas while reducing data density in other regions. This approach accelerates the training stage and enhances prediction quality in critical areas.

The successive refinement approach presented by Jahangirian and Johnston (1996) is used for the unstructured grid generation that is capable of producing high-quality (regular) stretched cells inside the boundary and shear layers as well as isotropic cells outside these regions. The compressible Navier-Stokes equations are solved using a finite volume cell-centered implicit scheme that follows the work of Jahangirian and Hadidoolabi (2005).

The CFD process is initiated using 525 pairs of Mach number and angle of attack, as depicted in Fig. 3. The selection of the number of inputs for the Mach number and angle of attack is based on experience and compatibility with the machine's structure during the dataset preparation stage.

To ensure the quality of the training process, it is important to select a portion of the available data as validation data. The validation data is used to assess the network's performance at the end of each epoch during training. Additionally, testing data is used to evaluate the network's prediction quality on data that was not encountered during the training process.

In this study, 80% of the total input data (corresponding to 420 flow fields) was utilized as training data. Validation and testing data were allocated as 17% (90 flow fields) and 3% (15 flow fields) of the total data, respectively. The total amount of data processed during training is calculated as follows: 420 (sets of Mach number and angle of attack) × 4 (pressure, u velocity, v velocity, and kinetic energy) × 5436 (number of computational cells) = 9,132,480. This number is significantly lower compared to the data requirements of image-based methods.

Fig. 5 MLP schematic structure

3. MULTI-LAYER PERCEPTRON (MLP) TRAINING

3.1 MLP Structure

The Multi-Layer Perceptron (MLP) is a type of neural network that comprises interconnected neurons (also known as node). It facilitates the mapping of inputs to outputs by passing data through layers within the network, starting from the initial layer and progressing towards the final layer. The output of each neuron is transmitted to the subsequent layer through weighted connections. During the training process, the weights are continuously adjusted until reaching their optimal values.

In the proposed method, the direct utilization of CFD data obviates the need for a complex network structure, making a simple MLP sufficient. Figure 5 illustrates the MLP structure employed in the current approach, which consists of three hidden layers, each containing 400 neurons. The input layer encompasses the Mach number and angle of attack, while the output layer corresponds to the CFD results.

Due to the presence of nonlinear relationships in CFD data, this study employs a nonlinear activation function.

For nonlinear problems, an MLP typically consists of three layers: the input layer, hidden layer(s), and output layer. The complexity of the problem may require additional hidden layers. Increasing the number of hidden layers allows the neural network to capture more intricate aspects of the problem. However, surpassing a certain threshold of layers can lead to increased training time without additional accuracy gains.

During the training process, the weights connected to each neuron are updated, resulting in changes to the value of the neuron. The output of the kth c Equation 5.

$$Y_k = \sigma\left(\sum_{i=1}^{n} (\text{weight}_{ki} \times \text{input}_i) + b_k\right) \tag{5}$$

Where Y represents the value of each neuron, σ denotes the activation function, b represents the bias, and input corresponds to the input dataset. The indices i and k indicate the position of two neurons connected by a weight, while n represents the total number of neurons in each layer.

For this study, the Rectified Linear Unit (ReLU) activation function has been employed due to its adaptability to the problem at hand. ReLU is a nonlinear function that effectively prevents excessive neuron activation, thereby simplifying the neural network structure and accelerating the training process.

The weights and biases of the neurons are updated using the back-propagation algorithm during the training phase (Rumelhart et al., 1986). This algorithm aims to minimize the Mean Square Error (MSE) defined in Equation 6 through an optimization process. The MSE is used to measure the difference between the predicted values generated by the neural network and the actual CFD data throughout the entire computational domain.

$$MSE = \frac{1}{n}\sum_{i=1}^{n} (E_i\text{-}P_i)^2 \tag{6}$$

In Equation 6, n represents the total number of data points in each input, while E and P refer to the exact CFD data and the predicted values, respectively. For the optimization algorithm, Adam was selected due to its excellent performance when applied to CFD datasets (Ruder, 2016). Unlike traditional stochastic gradient descent methods, Adam calculates adaptive learning rates for different parameters by estimating the first and second moments of the gradient.

3.2 MLP Hyper Parameters Setting

The MLP hyperparameters play a crucial role in determining the quality of the machine learning process. These parameters include the number of layers, nodes, activation functions, optimizer, learning rate, and batch size. Selecting the optimal parameters is essential to ensure the machine performs well on the validation data. The choice of hyperparameters is dependent on the dataset, and changing the dataset may require adjusting them accordingly.

There are two main approaches for selecting hyperparameters. The first approach involves trial and error, where parameters are chosen based on the user's experience with the dataset. However, this method can be time-consuming as it requires testing different combinations of hyperparameters on the validation data. Alternatively, the second approach automates the process of finding optimal parameters. Common methods for this include grid search, random search, and Bayesian model-based optimization. Grid search exhaustively searches the entire parameter space, increasing the chances of finding optimal parameters. However, for large design spaces, the time required to find the optimal parameters can be significant. In such cases, random search can be used to randomly select parameter values within their search space.

Grid and random search methods lack previous evaluations and can often waste time evaluate inappropriate hyperparameters. In contrast, the Bayesian method selects optimal parameters based on previous evaluations, resulting in fewer evaluations of the true objective function (loss function). This method builds a probability model of the objective function, which is simpler than the original model, and uses it to find the best hyperparameter combination. The surrogate model selects

Table 1 MLP settings

No. layers	No. hidden neurons	BS	VS	Optimizer	AF
5	400	64	17%	Adam	ReLU

* BS: batch size, VS: validation size, AF: activation function

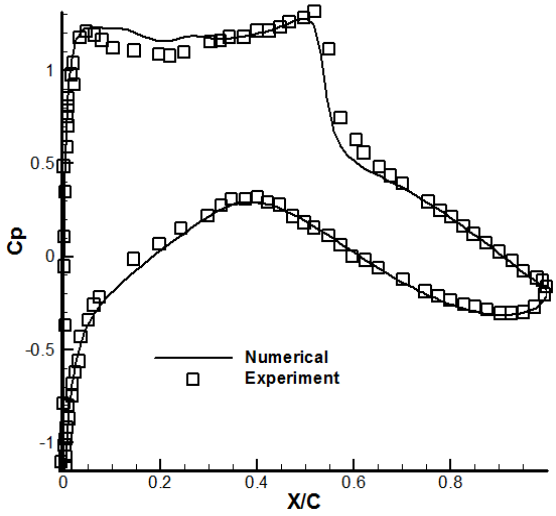


Fig. 6 Comparison of the surface pressure coefficients for the RAE2822 airfoil (case 9) from Dillmann et al. (2010)

promising parameters to evaluate the true objective function.

The advantage of the Bayesian method is that it requires less computational time to validate discovered hyperparameters due to the simpler structure of the probability model. The surrogate model is continuously updated to improve the probability model for selecting more appropriate hyperparameters.

In this study, due to the large design space, the hyperparameters of the neural network are automatically obtained using the Bayesian method. The Tree Parzen Estimators (Bergstra et al., 2011) and Expected Improvement (Jones et al., 1998) methods are used for constructing the surrogate model and evaluating hyperparameters, respectively. Table 1 presents the hyperparameter settings for the current neural network. It was found that normalizing the input and output data did not improve the training quality for this CFD-based dataset. Table 2 shows the optimization settings for Adam, where the learning rate can be increased to speed up training but may lead to convergence fluctuations, especially towards the end of the training process. The parameters β_1 and β_2 control the decay rates in the moving average calculations in Adam (see (Kingma & Ba, 2014) for further details).

Table 2 Adam settings for model training

Learning rate	β_1	β_2	ε
$1 \times e^{-3}$	0.9	0.999	$1 \times e^{-8}$

4. RESULTS AND DISCUSSIONS

The validation of the computational method for the RAE2822 airfoil in standard case number 9 has been performed, as shown in Fig. 6 and based on Dillmann et al., 2010. The flow conditions considered in this study are as follows: Mach number of 0.73, angle of attack of 2.79, and Reynolds number of 6.5 million. One prominent characteristic of transonic flow is the occurrence of shocks on the airfoil, and the results demonstrate a favorable agreement between the pressure distribution obtained from the numerical simulations and the experimental data.

In this study, the training process is conducted using TensorFlow (Abadi et al., 2016) with Keras (Chollet, 2015) running on top of Python's TensorFlow library. Using CFD-based data provides the advantage of using a simple MLP without the need for complex hardware during training, unlike image-based methods. Additionally, the number of datasets required is significantly reduced as the machine directly uses the data without the need for feature extraction operations present in convolutional methods. The training process in this study is performed on a Quad-Core AMD Opteron processor with a speed of 2.4 GHz, which is sufficient for training using the CFD-based method.

Figure 7 illustrates the convergence histories of MLP training and validation for all four flow fields. Figure 7(a) demonstrates the convergence of the training process for pressure after 4000 epochs. On the other hand, Fig 7(b) and 7(c) show that only 2500 epochs are needed for accurate prediction of velocity components U and V. Figure 7(d) indicates that the turbulent kinetic energy requires fewer epochs to achieve the minimum error compared to the other parameters.

The training process terminates at different epochs for each flow field due to their distinct nature. However, the achieved MSE for all trained networks is on the order of O (10^{-6}), indicating a highly accurate training process. The best results, indicated by the minimum validation loss, are saved during training to ensure that the optimal weights and biases are captured. This level of convergence error is crucial for accurate contours, especially within the boundary layer.

It is important to note that increasing the number of epochs does not always improve converged results and can lead to overfitting, where the machine becomes too focused on fitting the training data and fails to generalize well to new, unseen data. Determining the appropriate number of epochs for a given machine can be done through various methods, including trial and error.

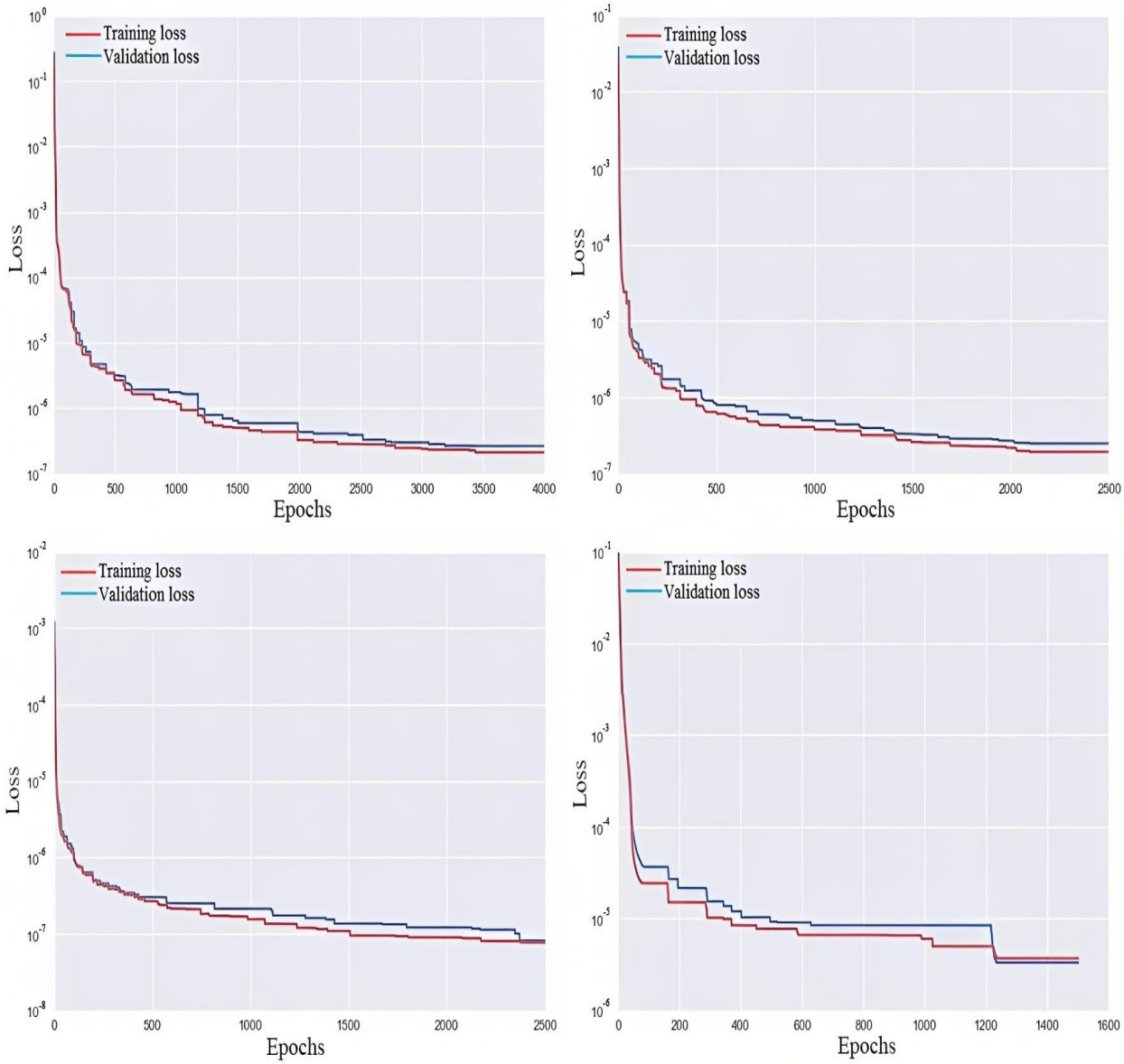


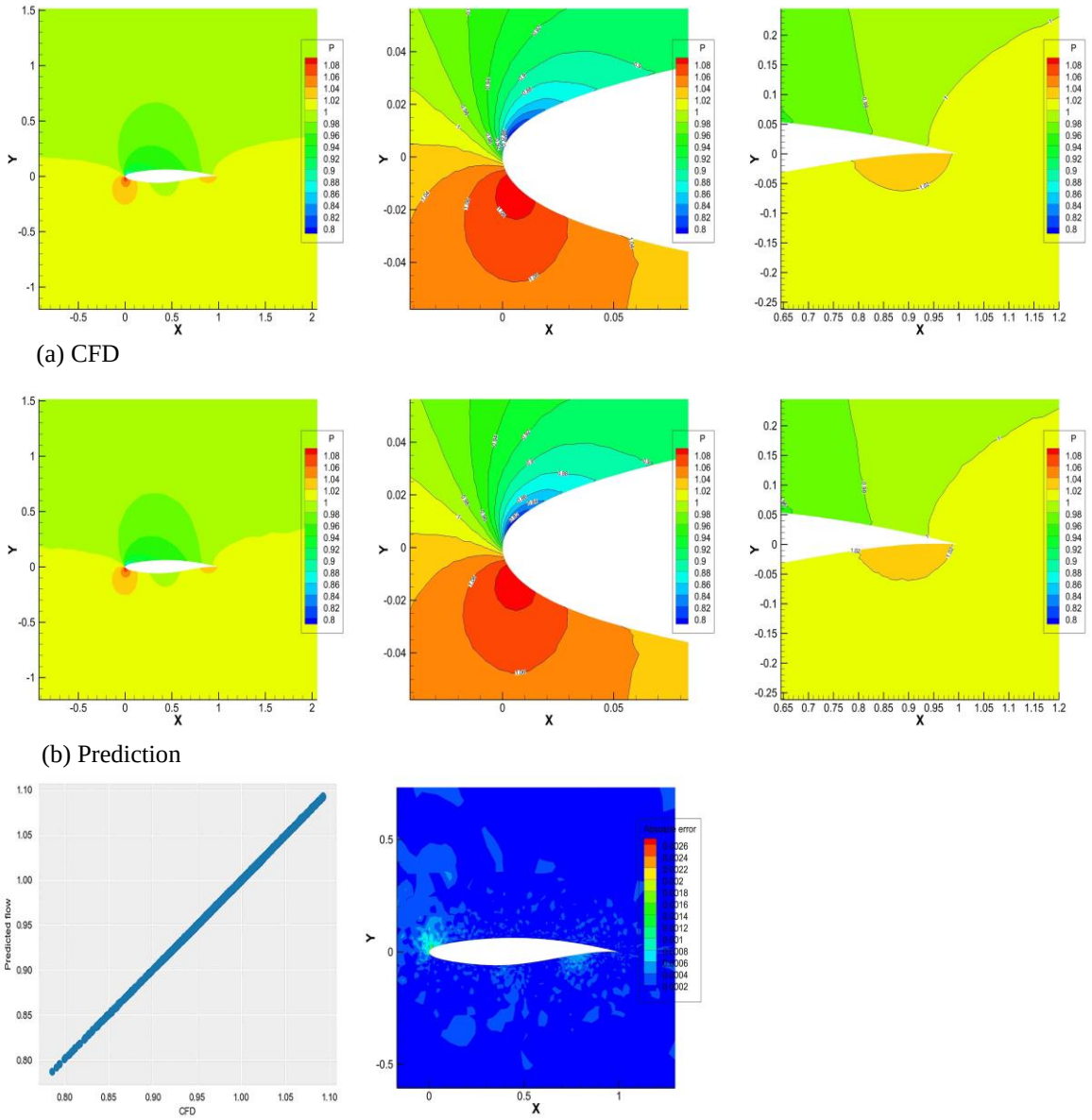
Fig. 7 Training and validation loss convergence for a) Pressure, b) U-velocity, c) V-velocity and d) turbulent kinetic energy

Table 3 Results for random testing cases

Flow field	Mach number	AOA	CFD time (s)	Prediction time	MSE (10^{-6})	Speed up
P	0.36	3.62	698	3.00	0.092	232
U	0.27	3.5	709	3.00	0.396	260
V	0.44	2.41	624	3.00	0.612	260
K	0.45	4.00	745	3.00	0.112	260

Table 3 presents the prediction results for all flow fields using a randomly selected sample from the testing cases. In all cases, the computational time for flow field prediction is 3 seconds, which is approximately 250 times faster than traditional CFD. The total training time for each flow field ranges from 6.9 to 12.3 hours. This significantly shorter training time is attributed to the simplified machine structure and the reduced size of the training dataset compared to image-based methods. The results indicate that, with the current neural network, having a larger number of inputs (beyond the 525 CFD flow fields) does not necessarily lead to better training outcomes but rather increases the training time. Conversely, reducing the number of inputs compromises the quality of training and prediction.

Figure 8 presents the contour plots of the pressure flow field obtained from both CFD and the neural network for a randomly selected testing case. The comparison of



(a) CFD

(b) Prediction

(c) Regression line (d) Absolute error

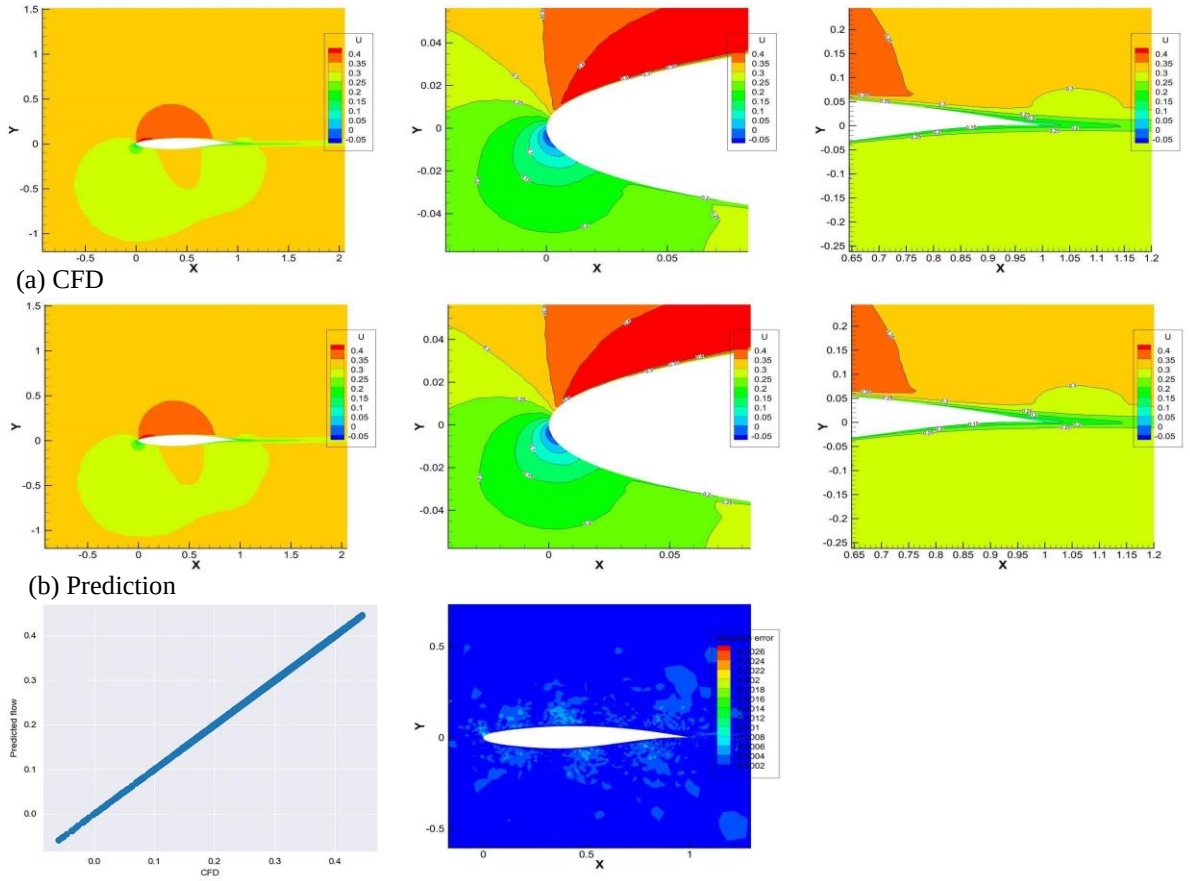
Fig 8 Pressure prediction for random testing case (M=0.36, AOA=3.62)

contour line patterns in Fig. 8(a) and 8(b) confirms the accurate prediction of the pressure flow field. Some small fluctuations can be observed in the contour lines, particularly behind the airfoil.

However, the close overlap between the regression results in Fig. 8(c) indicates that the prediction was made with high accuracy. The regression lines are used to validate the predicted and exact solution points. Since the predicted values are very close to the CFD results, the graph appears as a straight line.

Figure 8(d) illustrates the distribution of the absolute error between the CFD and prediction across the entire computational domain. Due to the rapid flow changes, such as the presence of a stagnation point, a small difference is observed near the leading edge. However, the trained model accurately predicts the values in this region, with an error of approximately 0.18% compared to the free flow pressure, which is negligible. Figures 9(a) and 9(b) demonstrate the close agreement between the predicted and CFD values, particularly for the U-velocity, as indicated by the convergence results in Fig. 7(b) with a minimal error. Figure 9(c) demonstrates a close alignment between the CFD and predicted results, indicating a low MSE.

To assess the performance of the trained machine in comparison to the CFD results, the U-velocity profiles at three specific points are examined: the leading edge, the middle (upper surface), and the wake region. Figure 10 presents the U-velocity profiles at x=0, 0.5, and 1.2 of the chord length. The U profiles obtained from both CFD and the prediction for the testing cases exhibit complete agreement.

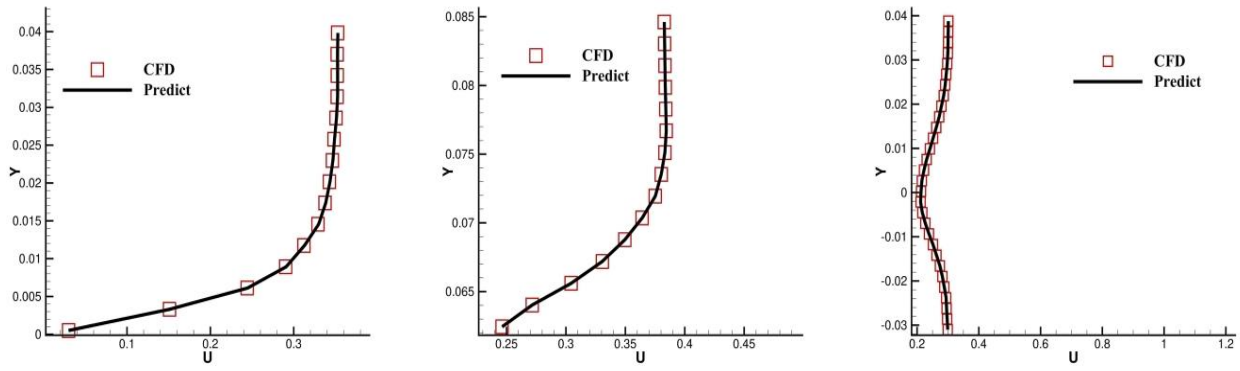


(a) CFD

(b) Prediction

(c) Regression line (d) Absolute error

Fig. 9 U-velocity prediction for random testing case (M=0.27, AOA=3.5)



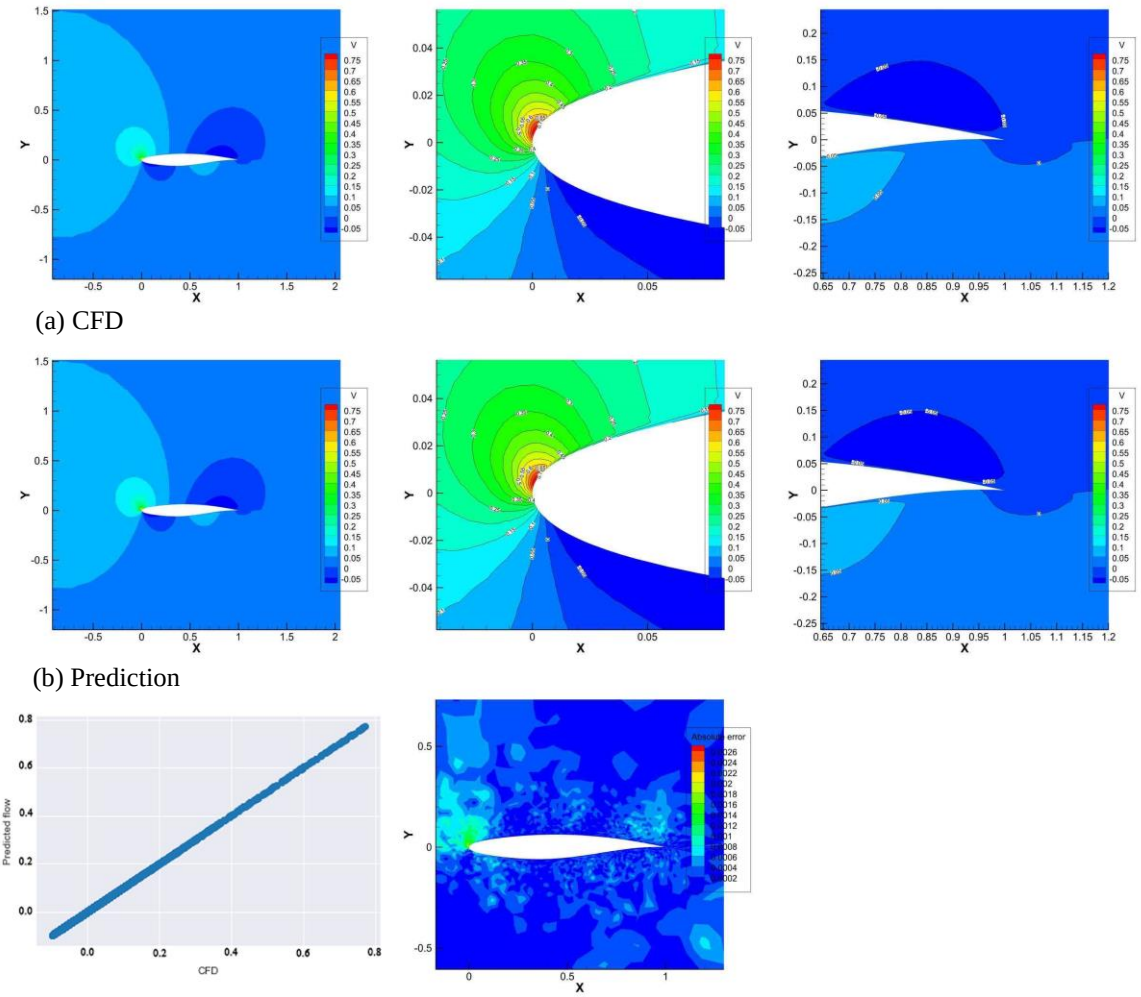
(a) x=0 (b) x=0.5 (c) x=1.2

Fig. 10 U-velocity profile at x=0, 0.5 and 1.2

Despite a slightly higher MSE error in the velocity components, the comparison between the CFD and predicted results reveals a strong similarity. The predicted velocity field exhibits a significantly lower error compared to the pressure field near the leading edge.

Figure 11 shows the prediction of the V-velocity flow field for a random testing case. The presence of accurate and smooth contour lines near the boundary layer suggests that the neural network performs well in terms of generalization. Figures 11(a) and 11(b) illustrate the similarity between the CFD and predicted flow fields.

In Fig. 12, the V-velocity profiles at x=0.0, 0.5, and 1.2 of the chord length are presented. At x=0.5, there is a slight deviation between the prediction and CFD results. However, the maximum error is only around 3%. The proposed method exhibits accurate predictions in critical regions such as the boundary layer, where the flow undergoes rapid changes.

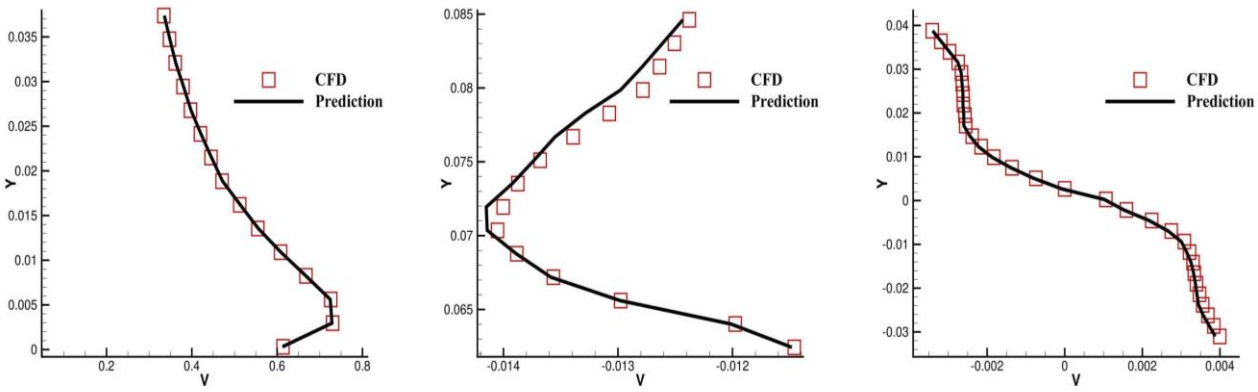


(a) CFD

(b) Prediction

(c) Regression line (d) Absolute error

Fig. 11 V-velocity prediction for random testing case (M=0.44, AOA=2.41)



(a) x=0 (b) x=0.5 (c) x=1.2

Fig. 6 V-velocity profile at x=0, 0.5 and 1.2

The computational grid is refined near these critical areas, making it suitable for machine learning. Figure 13 depicts the distribution of turbulent kinetic energy in the flow field. for a random testing case. Significant variations in kinetic energy are observed around the airfoil, particularly in the wake region. The presence of vortices near the trailing edge leads to higher kinetic energy in that area.

Due to the finer computational grid and the abundance of data near the airfoil, the prediction accuracy is significantly improved and closely resembles the CFD results.

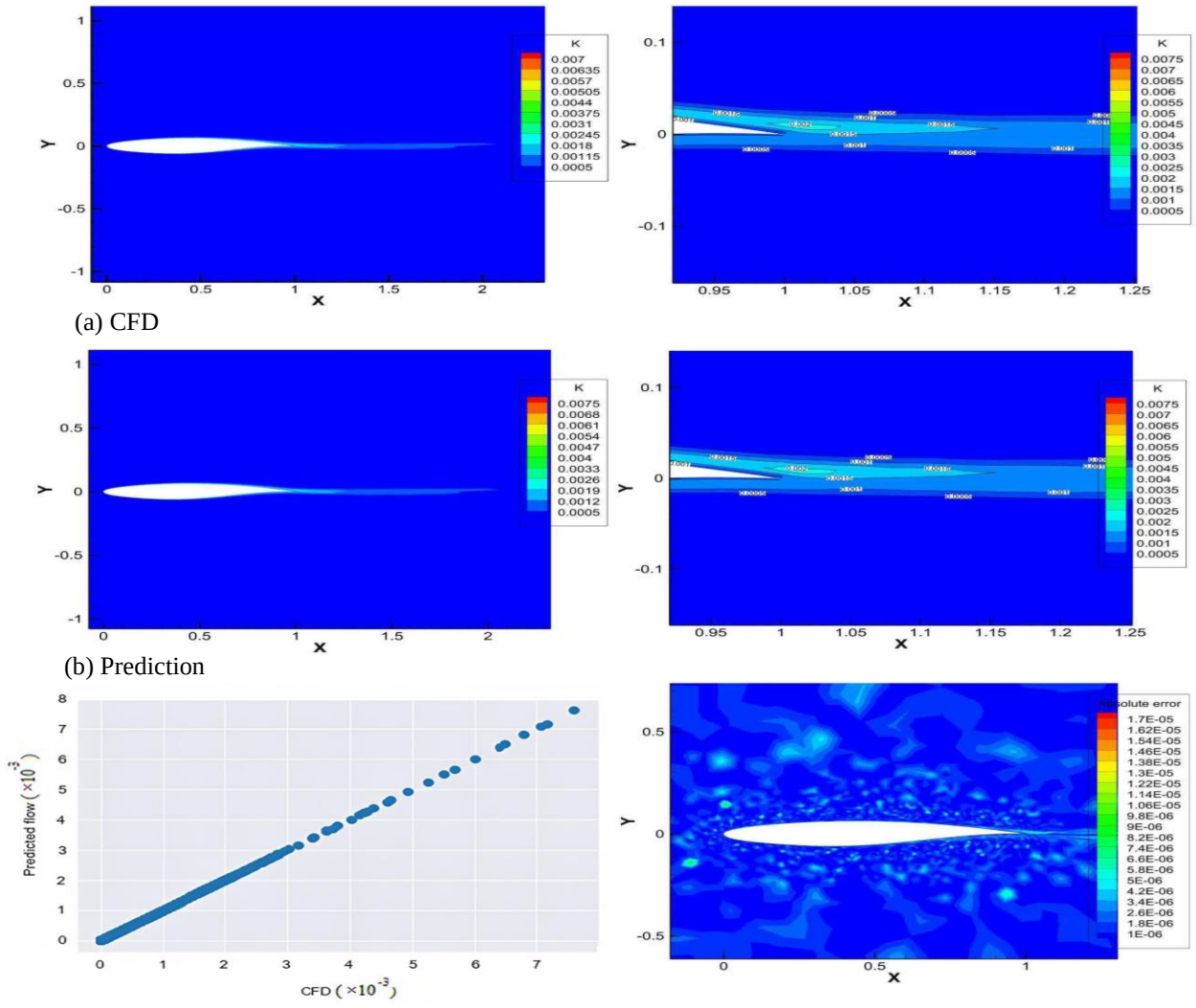


Fig. 7 Turbulence kinetic energy prediction for random testing case (M=0.45, AOA=4.00)

The obtained results demonstrate the benefits of utilizing direct CFD information in the machine learning process, as it allows for the processing of real data without introducing artificial features. In contrast, image-based methods often impose artificial characteristics on the neural network. Additionally, the Cartesian perspective of image-based methods treats all regions of the flow field equally, while the direct CFD approach allows for adaptive allocation of data based on importance. Another advantage of the direct CFD method is the ability to train the machine using conventional hardware, eliminating the need for powerful GPUs.

In future research, our method will be extended to predict the flow field for a morphing airfoil with variable flow and geometry conditions, particularly focusing on landing and takeoff scenarios (Nemati & Jahangirian, 2020). It is worth noting that morphing geometries introduce significant changes to the computational grid around the object, which can complicate the training process. Therefore, the selection of an appropriate dataset that accurately represents the main characteristics of the problem becomes increasingly crucial in addressing this challenge.

5. CONCLUSIONS

An efficient approach is proposed in this study to predict turbulent flow fields by utilizing a data-driven method based on direct computational fluid dynamics (CFD) inputs. Unlike traditional approaches that rely on flow field images, this method trains the machine using the values of computational grid cells. This enables a higher utilization of data in critical regions, thereby enhancing the accuracy of predictions. Conversely, in regions where flow variations are negligible, larger cells with lower data density are employed, without compromising the quality of predictions.

Neural network training is then conducted using the input data to predict the relevant fields of pressure, horizontal and vertical velocities, and turbulent kinetic energy.

Remarkably, this approach achieves a significant reduction of 88% in the amount of data required for training, leading to a simplified machine structure and lower training computational cost.

Consequently, it becomes possible to train the machine using conventional hardware instead of relying on powerful GPU systems.

Moreover, the convergence of training and testing loss reached an error magnitude on the order of O (10^{-6}) for all flow field variables, affirming the reliability of predictions.

In terms of computational efficiency, the time required for flow field prediction was less than 3 seconds, which represents a remarkable reduction of 260 times compared to the time needed with the CFD solver.

It is noted that the present method is applied for constant geometry and variable flow conditions, but it has the potential to encompass variable geometry and variable flow scenarios that is now under development.

CONFLICT OF INTEREST

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

AUTHORS CONTRIBUTION

All authors whose names appear on the submission made substantial contributions to the conception or design of the work and approved the version to be published.

REFERENCES

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., & Devin, M. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *ArXiv Preprint ArXiv:1603.04467*. https://doi.org/10.48550/arXiv.1603.04467

Akbıyık, H., & Yavuz, H. (2021). Artificial neural network application for aerodynamics of an airfoil equipped with plasma actuators. *Journal of Applied Fluid Mechanics*, *14*(4), 1165–1181. https://doi.org/10.47176/jafm.14.04.32133

Ansari, A., Mohaghegh, S., Shahnam, M., Dietiker, J. F., & Li, T. (2018). *Data driven smart proxy for cfd application of big data analytics & machine learning in computational fluid dynamics, report two: Model building at the cell level*. National Energy Technology Laboratory (NETL), Pittsburgh, PA, Morgantown. https://doi.org/10.2172/1431303

Bergstra, J., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for hyper-parameter optimization. *Advances in Neural Information Processing Systems*, 24.

Bhatnagar, S., Afshar, Y., Pan, S., Duraisamy, K., & Kaushik, S. (2019). Prediction of aerodynamic flow fields using convolutional neural networks. *Computational Mechanics*, *64*(2), 525–545. https://doi.org/10.1007/s00466-019-01740-0

Brunton, S. L., Noack, B. R., & Koumoutsakos, P. (2020). Machine learning for fluid mechanics. *Annual Review of Fluid Mechanics*, *52*, 477–508. https://doi.org/10.1146/ANNUREV-FLUID-010719-060214

Chollet, F. (2015). *Keras: deep learning library for theano and tensorflow*. URL: https://Keras.io

Dillmann, A., Heller, G., Schröder, W., Nitsche, W., Klaas, M., & Kreplin, H. P. (2010). *New Results in Numerical and Experimental Fluid Mechanics VII: Contributions to the 16th STAB/DGLR Symposium Aachen, Germany* (Vol. 112). Springer Science & Business Media. https://doi.org/10.1007/978-3-319-03158-3

Du, X., He, P., Technology, J. M. A. S. (2021). Rapid airfoil design optimization via neural networks-based parameterization and surrogate modeling. *Aerospace Science and Technology*, *113*. https://doi.org/10.1016/j.ast.2021.106701

Ghoreyshi, M., Jirasek, A., & Cummings, R. M. (2013). Computational approximation of nonlinear unsteady aerodynamics using an aerodynamic model hierarchy. *Aerospace Science and Technology*, *28*(1), 133–144. https://doi.org/10.1016/j.ast.2012.10.009

Guo, X., Li, W., Sigkdd, F. I. P. (2016). Convolutional neural networks for steady flow approximation. *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp. 481-490. https://www.osti.gov/biblio/416544*

Hallock, J. N., & Holzäpfel, F. (2018). A review of recent wake vortex research for increasing airport capacity. *Progress in Aerospace Sciences*, *98*, 27–36. http://dx.doi.org/10.1016/j.paerosci.2018.03.003

Hasegawa, K., Fukami, K., Murata, T., & Fukagata, K. (2020). CNN-LSTM based reduced order modeling of two-dimensional unsteady flows around a circular cylinder at different Reynolds numbers. *Fluid Dynamics Research*, *52*(6), 65501. http://dx.doi.org/10.1088/1873-7005/abb91d

Jahangirian, A., & Hadidoolabi, M. (2005). Unstructured moving grids for implicit calculation of unsteady compressible viscous flows. *International Journal for Numerical Methods in Fluids*, *47*(10–11), 1107–1113. https://doi.org/10.1002/FLD.877

Jahangirian, A. R., & Johnston, L. (1996). *Automatic generation of adaptive unstructured grids for viscous flow applications. 5th International Conference on Numerical Grid Generation in CFD, No. CONF-960489, Mississippi State Univ.* https://www.osti.gov/biblio/416544

Jin, X., Cheng, P., Chen, W. L., & Li, H. (2018). Prediction model of velocity field around circular cylinder over various Reynolds numbers by fusion convolutional neural networks based on pressure on

the cylinder. *Physics of Fluids*, *30*(4). https://doi.org/10.1063/1.5024595

Jones, D. R., Schonlau, M., & Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, *13*(4), 455. https://doi.org/10.1023/A:1008306431147

Kashefi, A., & Mukerji, T. (2022). Physics-informed PointNet: A deep learning solver for steady-state incompressible flows and thermal fields on multiple sets of irregular geometries. *Journal of Computational Physics*, *468*, 111510. https://doi.org/10.1016/j.jcp.2022.111510

Kashefi, A., Rempe, D., & Guibas, L. J. (2021). A point-cloud deep learning framework for prediction of fluid flow fields on irregular geometries. *Physics of Fluids*, *33*(2). https://doi.org/10.1063/5.0033376

Kavitha, R., & Mukesh Kumar, P. C. (2018). A comparison between MLP and SVR models in prediction of thermal properties of nano fluids. *Journal of Applied Fluid Mechanics*, *11*(Special Issue), 7–14. https://doi.org/10.36884/jafm.11.SI.29411

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *ArXiv Preprint ArXiv:1412.6980*. https://doi.org/10.48550/arXiv.1412.6980

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84–90. https://doi.org/10.1145/3065386

Li, Y., Chang, J., Kong, C., & Wang, Z. (2020). Flow field reconstruction and prediction of the supersonic cascade channel based on a symmetry neural network under complex and variable conditions. *AIP Advances*, *10*(6), 65116. http://dx.doi.org/10.1063/5.0008889

Marshall, J., Adcroft, A., Hill, C., Perelman, L., & Heisey, C. (1997). A finite-volume, incompressible Navier Stokes model for studies of the ocean on parallel computers. *Journal of Geophysical Research: Oceans*, *102*(C3), 5753–5766. https://doi.org/10.1029/96JC02775

McKay, M. D., Beckman, R. J., & Conover, W. J. (2000). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, *42*(1), 55–61. https://doi.org/10.2307/1268522

Miyanawala, T. P., & Jaiman, R. K. (2019). *A hybrid data-driven deep learning technique for fluid-structure interaction.* International Conference on Offshore Mechanics and Arctic Engineering, 58776, V002T08A004. https://doi.org/10.1115/OMAE2019-95870

Moureau, V., Domingo, P., & Vervisch, L. (2011). Design of a massively parallel CFD code for complex geometries. *Comptes Rendus Mécanique*, *339*(2–3), 141–148. https://doi.org/10.1016/j.crme.2010.12.001

Nagawkar, J., & Leifsson, L. (2022). Multifidelity aerodynamic flow field prediction using random forest-based machine learning. *Aerospace Science and Technology*, *123*, 107449. https://doi.org/10.1016/j.ast.2022.107449

Nemati, M., & Jahangirian, A. (2020). Robust aerodynamic morphing shape optimization for high-lift missions. *Aerospace Science and Technology , 103, 105897.* http://dx.doi.org/10.1016/j.ast.2020.105897

O'Shea, K., & Nash, R. (2015). An introduction to convolutional neural networks. *ArXiv Preprint ArXiv:1511.08458*. https://doi.org/10.48550/arXiv.1511.08458

Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). *Pointnet: Deep learning on point sets for 3d classification and segmentation.* Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 652–660. https://doi.org/10.48550/arXiv.1612.00593

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *ArXiv Preprint ArXiv:1609.04747*. https://doi.org/10.48550/arXiv.1609.04747

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533–536. https://doi.org/10.1038/323533a0

Sekar, V., Jiang, Q., Shu, C., & Khoo, B. C. (2019). Fast flow field prediction over airfoils using deep learning approach. *Physics of Fluids*, 31(5). https://doi.org/10.1063/1.5094943

Thuerey, N., Weißenow, K., Prantl, L., & Hu, X. (2019). Deep learning methods for reynolds-averaged navier--stokes simulations of airfoil flows. *AIAA Journal*, 1–12. https://doi.org/10.2514/1.j058291

Wang, Z., Liu, X., Yu, J., Wu, H., & Lyu, H. (2023). A general deep transfer learning framework for predicting the flow field of airfoils with small data. *Computers & Fluids*, *251*, 105738. http://dx.doi.org/10.1016/j.compfluid.2022.105738

Wu, H., Liu, X., An, W., Chen, S. (2020a). A deep learning approach for efficiently and accurately evaluating the flow field of supercritical airfoils. *Chinese Journal of Aeronautics*. http://dx.doi.org/10.1016/j.compfluid.2019.104393

Wu, P., Sun, J., Chang, X., Zhang, W., Arcucci, R., Guo, Y., & Pain, C. C. (2020b). Data-driven reduced order model with temporal convolutional neural network. *Computer Methods in Applied Mechanics and Engineering*, *360*, 112766. https://doi.org/10.1016/j.cma.2019.112766

Yu, W., Zhao, F., Yang, W., & Xu, H. (2019). Integrated analysis of CFD simulation data with K-means clustering algorithm for soot formation under varied combustion conditions. *Applied Thermal Engineering*, *153*, 299–305. https://doi.org/10.1016/j.applthermaleng.2019.03.011

Yuan, Z., Wang, Y., Qiu, Y., Bai, J., & Chen, G. (2018). *Aerodynamic coefficient prediction of airfoils with convolutional neural network.* Asia-Pacific International Symposium on Aerospace Technology, Springer Singapore. http://dx.doi.org/10.1007/978-981-13-3305-7_3