Contents lists available at http://qu.edu.iq

Al-Qadisiyah Journal for Engineering Sciences

Journal homepage: https://qjes.qu.edu.iq

Aerodynamic lift coefficient prediction of supercritical airfoils at transonic flow regime using convolutional neural networks (CNNs) and multi-layer perceptions (MLPs)

Olalekan Adebayo Olayemi[a,]*, Oluwadolapo Isaac Salako[a], Abdulbaqi Jinadu [a], Adebowale Martins Obalalu [b], and Benjamin Elochukwu Anyaegbuna [c]

[a]*Department of Aeronautics and Astronautics, Faculty of Engineering and Technology, Kwara State University, Malete, Nigeria.*
[b]*Department of Mathematical Science, Augustine University, Illara-Epe, Nigeria.*
[c]*Department of Mechanical Engineering, Faculty of Engineering and Technology, University of Ilorin, Ilorin 240003, Nigeria*

ARTICLE INFO

Article history:
Received 06 February 2023
Received in revised form 15 April 2023
Accepted 18 May 2023

Keywords:
Aerodynamics
Lift Coefficient
Supercritical Airfoil
Transonic
Neural-network

ABSTRACT

Designing an aircraft involves a lot of stages, however, airfoil selection remains one of the most crucial aspects of the design process. The type of airfoil chosen determines the lift on the aircraft wing and the drag on the aircraft fuselage. When a potential airfoil is identified, one of the first steps in deciding its optimality for the aircraft design requirements is to obtain its aerodynamic lift and drag coefficients. In the early stages of trying to select a candidate airfoil, which a whole part of the design process rests on, the conventional method for acquiring the aerodynamic coefficients is through Computational Fluid Dynamics Simulations (CFDs). However, CFD simulation is usually a computationally expensive, memory-demanding, and time-consuming iterative process; to circumvent this challenge, a data-driven model is proposed for the prediction of the lift coefficient of an airfoil in a transonic flow regime. Convolutional Neural Networks (CNNs) and Multi-Layer Perceptrons (MLPs) were used to develop a suitable model which can learn a set of usable patterns from an aerodynamic data corpus for the prediction of the lift coefficients of airfoils. Findings from the training revealed that the models (MLPs and CNNs) were able to accurately predict the lift coefficients of the airfoil.

1. Introduction

Over the past few years, the rapid development of artificial intelligence, specifically neural networks (NN) and several machine learning algorithms has fostered the growth and application of some prediction methods in fluid mechanics. Several airfoils have been developed and designed to date each with its aerodynamic coefficients that significantly affect flight performance and flight safety. The development of airfoils is an important aspect of aircraft aerodynamics. Aerodynamic forces (lift and drag) and motion are created by an airfoil-shaped body traveling through air. Lift is

* *Corresponding author.*
E-mail address: olalekan.olayemi@kwasu.edu.ng (Olalekan Adebayo Olayemi)

https://doi.org/10.30772/qjes.v16i2.955

the force that is perpendicular to the motion direction while drag refers to the force component that is parallel to the motion direction. An aerodynamicist will utilize airfoil shape to compute experimental lift, drag, and moment coefficients, as well as the pressure distribution, aerodynamic center, and other significant features during the airplane design stages. These computations are then extended to generate the aerodynamic properties of a finite wing so that its operation may be evaluated. As a result, aviation engineers place a high value on the numerical search for the best shape of airfoil/wing geometry.

A very crucial aspect of airfoil design is the calculation of the aerodynamic coefficients. The traditional systems of obtaining the aerodynamic coefficients are mainly through laboratory experiments and computational fluid dynamics [1]

The Computational simulation of deep learning models to gust detection in unsteady aerodynamics using surface pressure measurements from the wing surface was investigated [2]. The pioneering work about the convolutional neural networks (CNN) model was done by Sekar et al. [3] which defined the inverse design of airfoils. The model generated an airfoil geometry that showed a very similar pattern to the original airfoil with an error margin of less than 2% in most cases. Singh et al. [4] studied the viability of using deep neural networks to augment the Spalart-Allmaras turbulence model for better prediction of flow separations over airfoils. Their outcomes exhibited that when the CNN model generated is augmented with the Spalart-Allmaras model, the accuracy for the prediction of surface pressures improved significantly.

Yilmaz and German [5] presented results of the applications of deep learning for training predictors for airfoil performance. Their work centred around exploring the significance of deep learning over computational fluid dynamics simulations and surrogate modelling methods. Their deep learning approach was able to predict the aerodynamic coefficients of airfoils with over 80% accuracy when tested on real airfoil data.

Zhang et al. [6] adapted the CNN method for aerodynamic meta-modeling tasks involving variable flow conditions and geometry of an airfoil. Their newly proposed CNN technique was found to be comparable with existing Multi-Layer Perceptron (MLP) techniques in learning capabilities. Zhu et al. [7] studied machine learning models for low Reynolds (Re) number flows based on direct numerical simulation has been seen to give very high accuracy and generalization abilities; but the flow around the airfoils increases in Reynolds number, the machine learning models are unable to generalize and often leads to high inaccuracies due to scaling effects. Rehan and Javed [8] in their study, compared two techniques in predicting the aerodynamic coefficients of flow past a supersonic missile. CFD and Missile DATCOM were employed for the analysis and prediction of the aerodynamic coefficients for supersonic speeds of Mach 2, 3, and 4 with angles of attack varying from -20º to +20º. Their work showed that the values for the coefficient of lift obtained from CFD and Missile DATCOM were very similar for smaller angles of attack (less than 10°), however for large values of angles of attack (beyond 10°) the difference in the values increases. Also, the coefficient of drag increases smoothly at smaller angles of attack (less than 10°).

Viquerat and Hachem [9] investigated the application of deep learning of low Reynolds numbers on laminar flow regimes for predicting the drag coefficients of arbitrary 2D shapes.

Kutz [10] explored the various applications of deep learning techniques to fluid dynamics. He opined that turbulent flows exhibit multi-scale physics with the presence of rotational and translational intermittent structures. Santos et al. [11] examined the application of NN in forecasting aerodynamic coefficients of airfoils for varying Mach numbers and angles of attacks. They reported that the error for the drag and lift coefficients are still beyond practical use in multidisciplinary design and optimization applications, but the errors could reduce significantly by increasing the nodes and neurons of the network and training with a larger database.

Miyanawala and Jaiman [12] proposed an efficient deep-learning technique for the Navier-Stokes equation for the purpose of applying it to unsteady flow dynamics. Using the convolutional neural network and stochastic gradient descent methods, they were able to predict the force coefficients of bluff body geometries up to a relative error rate of less than 5%. Their results proved that it is possible to use deep learning methods to train the output of force coefficients for any perturbed bluff body geometry input.

Sun and Wang [13] in their investigation of Artificial Neural Network surrogate modelling (ANNSM) in aerodynamic design showed that ANNSM has an edge over other existing modelling methods in terms of economic computational capabilities and accurate generalization capabilities. Morimoto et al. [14] in their study of convolutional neural networks for fluid flow analysis posited that the convolutional neural network model is not only robust to dimension reduction operations but also sensitive to dimension extension methods. Li et al. [15] proposed a new sampling method for airfoils and wings to reduce abnormalities of both initial and infill samples based on a deep convolutional generative adversarial network. They observed that the network was able to generate sample airfoils that are notably more realistic than those generated by other sampling methods. Tompson et al. [16] studied the application of convolutional neural networks to accelerate Eulerian fluid simulation. They observed that the data-driven approach can solve the Navier-Stokes equation or fluid flow significantly faster than CFD methods. Wang et al. [17] proposed a data-driven shape encoding and generating method that can automatically learn existing airfoil geometries and use it to generate new airfoils. Their experiments showed that the model learns compact and comprehensive features encoding shape information of airfoils and can automatically generate novel airfoils. Haizhou et al. [18] presented a newer technique for data augmented Generative Adversarial Network (daGAN) that is capable of rapid and accurate flow field prediction, the model also readily adapts to tasks with sparse data. Duru et al. [19] presented a convolutional neural network system that works on the encoder-decoder technique for predicting the pressure fields formed around an airfoil.

Thirumalainambi and Bardina [20] demonstrated the efficiency and reliability of using neural networks to predict aerodynamic coefficients modelled as a function of angle of attack, speed brake deflection angle, Mach number, and sideslip angle. They also experimented on the number of training data points and the type of transfer function to be used between the input-hidden layer and the hidden-output layer required to produce an efficient neural network prediction. While proposing a different technique for predicting the aerodynamic coefficients of airfoils using a convolutional neural network (ConvNet) as well as a signed distance function (SDF), Yuan et al. [21] opined that one of the major downsides to traditional surrogate-based prediction methods is its limitations when it comes to the dimensions of design variables, and it is also inefficient for strong nonlinear engineering problems. Their results represent one of the early steps in the development of machine-learning-based aerodynamic coefficient prediction tools. Karali et al. [22] investigated the use of a deep learning-based surrogate model to determine the non-linear characteristics of unmanned aerial vehicles. It was demonstrated that the model is cable of predicting the aerodynamic properties of different unmanned aerial vehicle design parameters quickly by using only the geometric configuration parameters without the need for any special input data or pre-process phase.

Fukami et al. [23] assessed the use of supervised machine learning methods for fluid flows, after examining four machine learning architectures in terms of their accuracy, costs of computations, and robustness for flow problems. Hui et al. [24] studied the application of convolutional neural networks in predicting the pressure distribution over an airfoil. Given unseen airfoils from the validation dataset, their model predicted pressure coefficients in seconds and had less than 2% mean square error. Kim and Lee [25] investigated the application of convolutional neural networks in modelling turbulent heat transfer.

Sun et al. [26] developed a deep learning-based prediction approach for supercritical airfoils at transonic speeds. The methods used in the study require the geometric parameters of the airfoil which are then represented and parameterized with various shape functions. The present methodology feeds the airfoil images directly into the prediction model as input and is therefore commonly referred to as the graphical prediction method. Compared with the parametric method, there are three advantages. First, the airfoil image can accurately represent the airfoil geometry. Second, the inputs of the prediction model are airfoil images and flow conditions, and no geometric parameters of the airfoil are needed. Third, the training time of the prediction model increases linearly with the number of samples.

A Multi-Input Network aerodynamic-coefficient prediction method of airfoils based on CNN and MLP is proposed in this paper. This method belongs to the graphical method, when compared with the work of Sun et al. [26], there are two differences, first the number and types of input and output of the prediction models are different. The prediction model established in this paper is a multi-input and single-output regression. The input includes the airfoil image and flow condition, and the output is only the lift coefficient CL. Second, in this paper, the airfoil image is tilted by the corresponding angle of attack, this process can keep the basic shape of the airfoil and makes it comfortable for CNN to extract image features. Hence, from the literature search conducted, no similar work employing the same methodology had been previously reported for supercritical airfoils under a transonic flow regime. Therefore, the current investigation attempts to bridge the identified gaps above.

2. Methodology

This research uses a deep learning approach to solve regression issues in which the algorithm is trained using labeled inputs to extract the data's underlying properties. The algorithms selected for this research are CNN Rikiya and Nishio [27] and MLP Marius-Constantin et al. [28] and there are 3 major advantages of utilizing CNN instead of other competing networks. First, there is very little dependence on pre-processing, decreasing the need for human effort to develop its functionalities. Second, it is easy to understand and fast to implement. Third, it has the highest accuracy among all algorithms Rikiya and Nishio [27]. The information on the MLP, CNNs, and the details of how the aerodynamic data was generated is explained succinctly.

2.1. Aerodynamic Data Generation

To create the aerodynamic coefficient data used in this work, it was necessary to calculate by means of computation the velocity and pressure distribution of flows surrounding airfoils. These simulations were performed over a wide range of Mach numbers ($0.94 \leqslant Ma \leqslant 1.4$) and angles of attack in the range of ($-20^\circ \leqslant \alpha \leqslant 20^\circ$). The input data was created by taking random samples of freestream circumstances from the range specified above in conjunction with ten (10) distinct airfoil shapes taken from the UIUC database (Selig, 1996). The RANS simulations were carried out using the k SST two-equation turbulence model, and the solutions were derived using the widely used ANSYS Fluent Computational Fluid Dynamics Software. The mesh used in this calculation is a body-fitted triangular mesh with refinement near the airfoil.

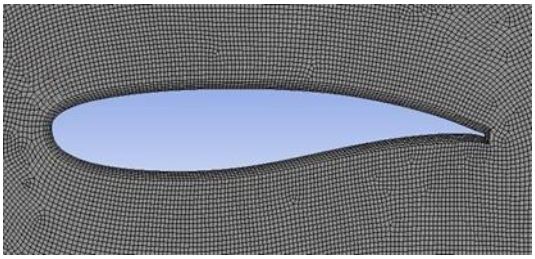


Figure 1. Body Fitted Triangle Mesh Around Airfoil

Parameters such as Reynolds number, angle of attack, and airfoil were randomly chosen from the aforementioned ranges to run the simulation. Since the learning algorithm relies on this data, the dataset was split into a training set, test set, and validation set. A 70/15/15 split was used for training, testing, and validation sets respectively. Testing sets are a good way to find out if the trained model has any bias. If it is overfitting, the model is fine-tuned and retrained on the training set. The validation data set that was used comprises 15% of the dataset that was not included in the training and testing data. To assess the trained models' accuracy, parameters such as Reynolds number, angle of attack, and airfoil were randomly chosen from the parameter ranges earlier specified to run the simulations. The final model's performance on the validation set is a representation of the model's performance when deployed for practical cases.

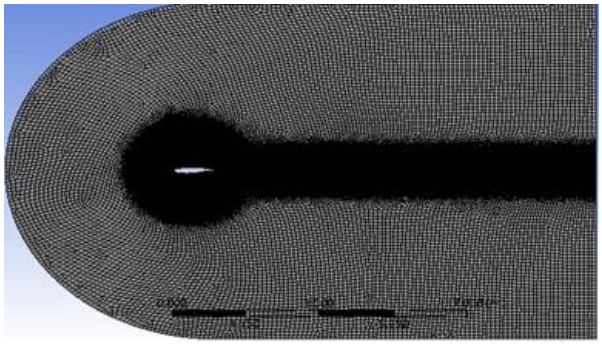


Figure 2. Full mesh around the airfoil.

2.2. Multilayer Perceptron (MLP)

The most popular kind of neural network is the multilayer perceptron type [28]. It is made up of numerous neurons that are intricately coupled to form a network [3]. MLP is a kind of perceptron that consists of many perceptrons. They are made up of an input layer that accepts incoming signals and an output layer that makes a prediction based on the input, with an arbitrary number of hidden layers in between that act as the MLP's true computational engine. MLPs may approximate any continuous function and can approximate any function with only one hidden layer.

Multilayer perceptrons are commonly employed in supervised machine learning; they learn to represent the relationship (or correlations) between a set of input-output pairs by training on them. Multilayer perceptrons are also used in unsupervised deep learning. The process of training involves

altering the parameters of the model, such as the weights and biases, to decrease error. Backpropagation is used to conduct such weight and bias modifications in proportion to the error, and the error may be measured using a variety of ways, including the root mean squared error (RMSE).
The power of the multi-layer perceptron (MLP) network stems precisely from non-linear activation functions, of which neurons are the most significant elements. This technique works well with most non-linear functions, excluding polynomial functions. The sigmoid function in equation (1) is the most prevalent operation.

$$f(s) = \frac{1}{1 + e^{-s}} \quad (1)$$

- **Linear Threshold Units (LTUs)**

Multilayer perceptron networks are built from linear threshold units (see Figure 3), which typically consist of a single-value input x with n values and a single-value output y, with some mathematical operations in between to estimate the linear function of the inputs and their scales and use the activation function to evaluate the mathematical operation [29].

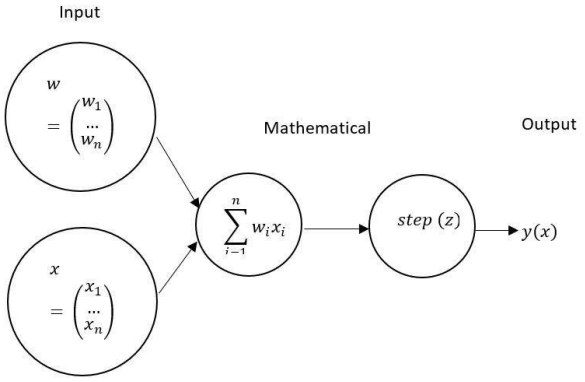


Figure 3. Linear Threshold Unit (LTU)

- **Multilayer Perceptions (MLPs)**

A perceptron is a basic artificial neural network with a single layer of LTUs that is coupled to all the inputs and the bias vector. An input layer with at least one hidden layer of LTUs, and an output layer of LTUs make up a Multi-Layer Perceptron (MLP). A deep neural network (DNN) is one that has two or more hidden layers. The neuron contains n inputs x_1, x_2,…x_n and associated weights w_1,w_2,…w_n, as illustrated in Figure 4. The weighted input for MLP network is given by equation (2)

$$Weight\ input = \sum_{i=1}^{n} w_i x_i + b \quad (2)$$

The weighted input is called signal to the neuron z, where b is a bias term. The output is obtained by passing the input signal through a nonlinear activation function σ (·). As a result, the neuron output is expressed by equation (3):

$$a(x) = \sigma(\sum_{i=1}^{n} w_i x_i + b) = \sigma(z) \quad (3)$$

where a(x) is the activation output (i.e., the neuron output) and z denotes the neuron signal. Figure (4) depicts a typical MLP network. Where b indicates the bias term, sigma represents the activation function.

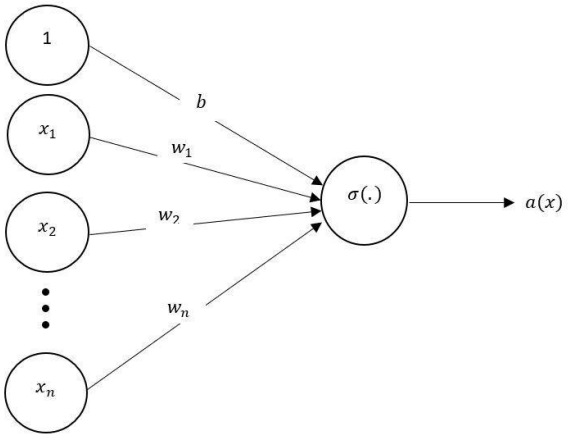


Figure 4. A typical MLP network with inputs [x1, x2 ... xn] and weights [w1, w2 ... wn]

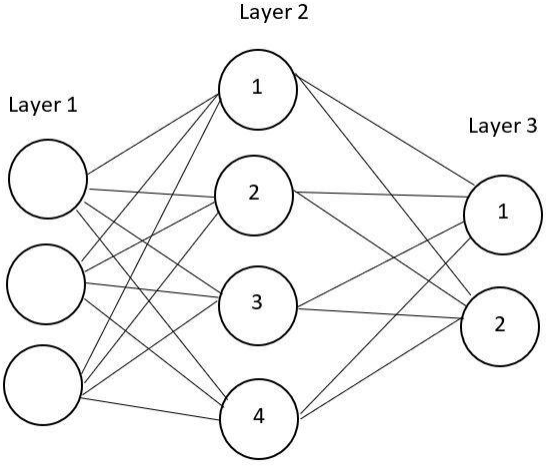


Figure 5. An MLP network with a multiple hidden layer

2.3. Convolutional Neural Networks (CNNs)

Each input (for example, a pixel in an image) is represented by a single perceptron in MLPs and the number of weights quickly becomes unmanageable for large images. Since it is fully connected, it has an excessive number of parameters. Each node is connected to every other node in the preceding and subsequent layers thus resulting in a dense web of connections that promotes system redundancy and inefficiencies [30]. As a result of this, issues may arise during training, and overfitting may develop which could impair the ability to generalize the results. Therefore, MLPs are not the optimal choice for image processing. One of the primary issues is that when a picture is flattened (matrix to vector), spatial information is lost. Thus, a method is required for exploiting the spatial correlation of the image features (pixels) in such a way that it is never distorted, which is why LeCun et al. [31] introduced the Convolutional Neural Networks which effectively addresses this issue. The CNN adopted for the current study is displayed in Figure 6.

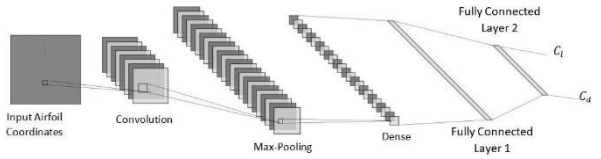


Figure 6. CNN architecture used in the current investigation.

CNNs are composed of a variety of layers some of which include the convolutional layer, the pooling layer, and the fully connected layer.

- **Convolutional layer**

Convolutional layers contain most of the computation and they are where CNN's processing focuses. An input data, a filter, and a feature map are required to make it work. Assuming the input is a 3D image made up of pixels; the input will contain three dimensions, matching Red Green, and Blue (RGB) image data.

A 2-D array of weights is the feature detector, which acts as a proxy for an image's portion. There are several different sizes a filter can take, but one of the most common ones is a 3x3 matrix, which affects the size of the receptive field. The activation map is formed of neurons and is created using convolution [32]. In other words, the filter slides across the input in both directions and computes dot products in every spatial position. Convolutional layers calculate the volume of their output by stacking activation maps in the depth dimension. The activation map contains many neurons, each of which only has connections to a small local portion of the input volume. Figure 7 is a graphical illustration of the convolution operation [33]. In conclusion, neurons have tiny receptive fields, which match the filter size.

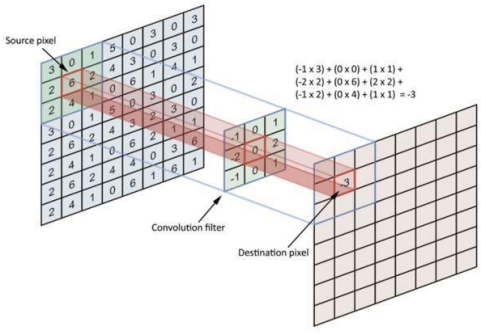


Figure 7. Typical convolution operation in the CNN.

- **Pooling layer**

In order to reduce the number of parameters, the data is transformed into smaller representations, and this is often accomplished using pooling layers. The pooling operation, like the convolutional layer, covers the entire input with a filter that has no weight, making it different from a convolutional layer [34]. The kernel appends input values by applying an aggregation function to input values. After each time the input moves, Maximum pooling assigns the greatest pixel value to the output array. A disadvantage of the pooling layer is that it introduces a considerable deal of information loss, yet it offers a few benefits to CNN which include benefits in limiting overfitting risk, enhancing efficiency, and reducing complexity.

- **Fully Connected layer (FC)**

A completely connected network relies on the input values having direct ties to the output layer. These input values are not directly linked to the output layer in partly connected layers. However, each node in the output layer is connected to a node in the preceding layer, whereas each node in the fully connected layer links towards another node in the same layer. The task of classification in this layer relies on features and different filters for features that were extracted by prior layers. FC layers normally use a softmax activation function, which is popularly employed for classifying inputs correctly and producing a probability between 0 and 1, while ReLu functions are used in convolutional and pooling layers [35].

3. Results and discussion

3.1. Multi-Input Network

To deal with the dataset's diversity, which comprises numerical and image data, two primary networks were built. To handle numerical inputs, the initial branch of the network uses a simple Multi-layer Perceptron (MLP) while a Convolutional Neural Network was used to process the image data in the second branch. The final multi-input Keras model was constructed by concatenating these branches. The Keras API is utilized by the MLP. MLP consists of the following:

•ReLU is used as the activation function in this fully linked input layer.
•A completely linked hidden layer that also employs ReLU to activate it.
•Dropout regularization of 0.5 was also added to prevent overfitting. Dropout is a term used in machine learning to describe the act of randomly disregarding certain nodes in a layer during training.
•The number of hidden neurons utilized for the study is 16.
•Lastly, a regression output with linear activation is given.

The network was developed to produce a model for the prediction of lift characteristics of airfoils The training database of the models comprises of eight (8) supercritical airfoils' geometries over a wide range of Mach numbers (0.94≤Ma≤1.4) and angles of attack in the range of (-20° ≤ α ≤20°), the respective Reynolds number was computed for each of the Mach numbers in the database. The CNN architecture is implemented, trained, and predicted using the open-source program Tensorflow via the Keras Application Programming Interface [36]. The backpropagation algorithm is applied to train the weights and biases of the CNN to minimize the defined mean square error (MSE). For a batch of n training samples, the mean square error (MSE) is calculated using equation (4).

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2 \quad (4)$$

Where y_i are the true data, and $\tilde{y}_i$ is the network prediction.

Additionally, the backpropagation technique computes gradients that are then optimized using the ADAM optimizer [37]. Adam is an optimization algorithm that, in place of the standard stochastic gradient descent procedure, can be used to iteratively update network weights using training data.

In Figure 8, an initial learning rate of 1.0 x 10^(-3) was selected and the learning rate scheduler reduced the learning rate in steps when the MSE of the validation set shows no significant improvement over several epochs

(i.e from the epoch of beyond 400). The weights and biases are adjusted after exposing the CNN to one batch of samples in minibatch mode training, with an initial batch size of 32.

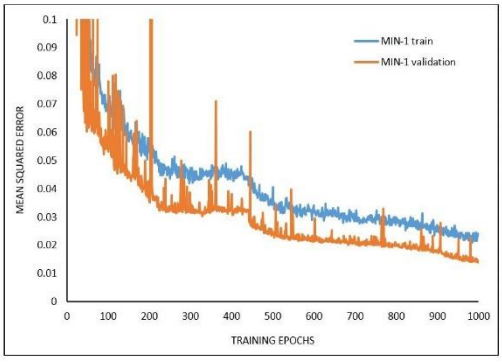


Figure 8. Mean Squared Error training and validation convergence history with an input image size of 108x72x2

3.2. Influence of Input Image Size

In Figure 9, the effects of the input image size of the airfoils were investigated. Two sizes of the input with dimensions 108 × 72 × 2 and 200 × 133 × 2 were investigated. The figures reveal that though the validation convergence is oscillatory, the plots show that the mean squared errors for both plots are minimized beyond 200 epoch training; therefore, the image size increase helps the convergence profiles. Also, the Mixed Input Network (MIN-2) comprising both the Convolutional Neural Networks and Multi-Layer Perceptron is proficient in operating in equal input sizes as revealed by the convergence errors.

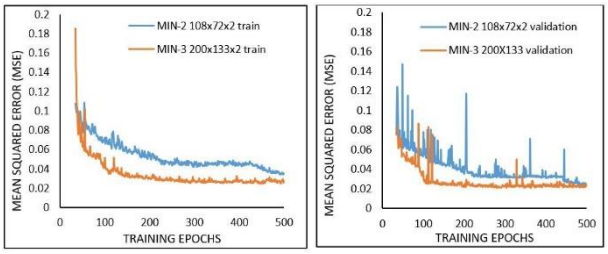


Figure 9. (a) Mean Squared Error (MSE) convergence history of MIN-2 with an image size of 108x72x2 training (left); (b) Mean Squared Error (MSE) convergence history of MIN-3 with an image size of 200x133x2. validation (right).

3.3. Influence of Training Parameters

- **Minibatch size**

The effects of batch size on convergence were also investigated on MIN-3. In this study, three different batch sizes 32, 64, and 128 were selected and their effects on training convergence are shown in Figure 10.

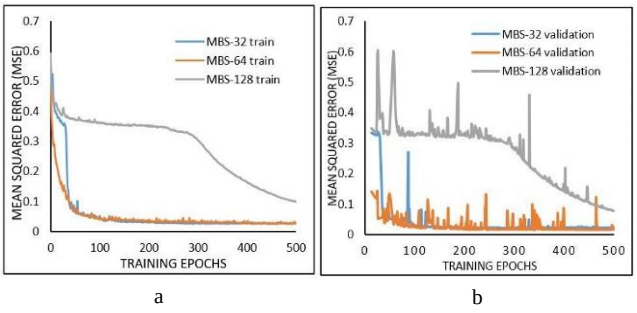


Figure 10. (a) MSE convergence history of MIN-3 with different minibatch training (left); (b) MSE convergence history of MIN-3 with different minibatch validation (right).

In Figure 10, MBSs of 32 and 64 outperformed MBSs of 128 in terms of convergence. As a result of the convergence histories, an MBS of 64 was selected as the best for training.

- **Learning rate**

The effects of varying the learning rates on the Mean Squared Error (MSE) convergence of MIN-3 are also investigated. Initial learning rates (LRs) of 1 × 10−2 and 1 × 10−5 were selected and its influence on this study is shown in Figure 11.

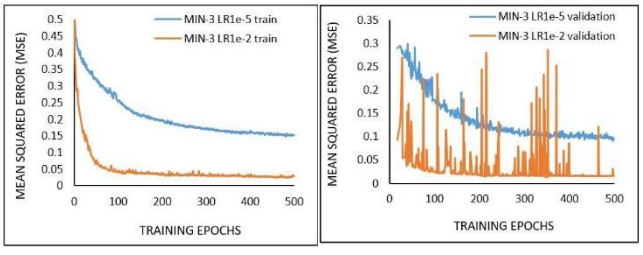


Figure 11. MSE convergence history of MIN-3 with varying learning rates.

As evidenced in Figure 11, choosing a larger learning rate results in oscillatory convergence, especially on the validation data, which shows an inability of the model to generalize and fit unseen data appropriately. Additionally, a learning rate of LR1e-5 was chosen due to its smoother convergence and acceptable error range.

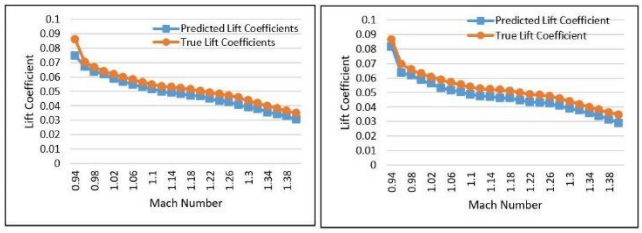
- **Network Prediction**

The trained MIN-3 architecture was utilized to forecast the airfoils in the training and testing sets. The testing dataset comprises airfoils that the network did not observe during training. Table 1 shows the equivalent needed computational time of MIN-3 for training and predictions.

Table 1. Network Training and Prediction Time for Min-3.

Multi-Input Network (Min) Type	CPU (Intel R Core i5 at 3.00GHz) Time (Hours)
MIN-3 Training (10332 samples)	8.5277778
MIN-3 Predictions (2952 samples)	6.16 x 10^(-3), 2.08 x 10^(-6) per sample

As demonstrated in Table 1, MIN-3 predicts the aerodynamic coefficients of a given airfoil in less than one second, and Table 2 shows the MSE for all the explored MIN architectures in this study. Fig. 12 depicts the performance of the training and testing samples. The plots show that the Multi-Input Network performed better in predicting the aerodynamic lift coefficients of any given supercritical airfoil.



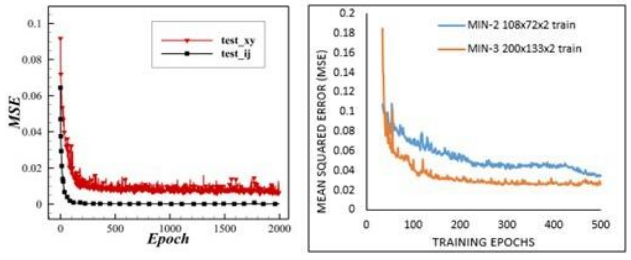
a b

Figure 12. (a) Fig. 12: MIN-3 (200 × 133 × 3) prediction of some airfoils at attack angles of 20° (left); (b) Fig. 12: MIN-3 (200 × 133 × 3) prediction of some airfoils at attack angles of 10° (right).

Table 2. MSE error on different MIN architectures for both training and validation. An MBS of 32 and LR of 1e-2 were used unless explicitly stated. (es = early stopping).

MIN Architecture	Training Error	Validation Error
MIN-1	0.021406367	0.014135924
MIN-3	0.024861962	0.014135924
MIN-3 MBS-64	0.024698643	0.015846226
MIN-3 MBS-128	0.097584948	0.078603454
MIN-3 LR1e-5 (es)	0.266813695	0.237168238
MIN-3 MBS-64 LR1e-5	0.151511803	0.095736757
MIN-3 MBS-128 LR1e-5 (es)	0.281726569	0.294390082

Fig.13 (left) presents the Mean Squared Error (MSE) from the work of Sun et al. [13] while Fig. 13 (right) displays the MSE plot from the present work. The trends of both plots are similar despite the difference in the method of geometry representation and algorithms.



a b

Figure 13. (a) Comparison of Sun et al. [26]; (b) present work.

4. Influence of input image size

In this study, a methodology for predicting airfoil aerodynamic coefficient based on Convolutional Neural Networks (CNN) and Multi-Layer Perceptron (MLP) was developed and effectively applied for the prediction of lift coefficients of supercritical airfoils in transonic flow regimes. The geometry of the airfoil was fed as input to the CNN, while parameters such as Mach Number, and angle of Multi-Input Reynolds Number were fed as input to the MLP. The joining of the CNN and MLP formed the Multi-Input Network (MIN) and the aerodynamic lift coefficients were obtained as output. Several MIN architectures (MIN-1, MIN-2, and MIN-3) with varying depths and network parameters were trained. This approach to estimating aerodynamic coefficients is faster and can easily be used to iterate through multiple airfoils to find the optimal coefficient for a specific application before performing the more accurate CFD simulations on the selected airfoils. It is obvious from the convergence error that the selected designs have comparable error convergence. Also, from the training carried out, the models (MLPs and CNNs) were able to accurately predict the lift coefficients of the airfoil. Furthermore, the impacts of the input image size, mini-batch size, and learning rate on the level of performance using the selected MIN-3 architecture were evaluated. The convergence error of MIN-3 with two different input image sizes (108 x72 x3 and 200x133x 3) shows that increasing the input image size does not enhance the results considerably. However, a small batch size of 64 and a learning rate of 1e-5 greatly improved the model. The level of accuracy predicted by the model used for the current investigation needs to be improved upon for it to be useful in aircraft design and optimization procedures.

5. Data availability statement

The authors confirm that the data needed to support the findings in this study are available within the article.

Authors' contribution

All authors contributed equally to the preparation of this article.

Declaration of competing interest

The authors declare no conflicts of interest.

Funding source

This study didn't receive any specific funds.

REFERENCES

[1] Chen, H., He, L., Qian, W., Wang, S. 2020. Multiple aerodynamic coefficient prediction of airfoils using a convolutional neural network. Symmetry, 12, 1-14.

[2] Hou, W., Darakananda, D., Eldredge, J. D. 2019. Machine-Learning-Based Detection of Aerodynamic Disturbances Using Surface Pressure Measurements. American Institute of Aeronautics And Astronautics Journal, 57, 5079–5093.

[3] Sekar, V., Zhang, M., Shu, C., & Khoo, B. C. 2019. Inverse design of airfoil using a deep convolutional neural network. American Institute Of Aeronautics And Astronautics Journal, 57, 993–1003.

[4] Singh, A. P., Medida, S., Duraisamy, K. 2017. Machine-learning-augmented predictive modeling of turbulent separated flows over airfoils. American Institute Of Aeronautics And Astronautics Journal, 55, 2215–2227.

[5] Yilmaz, E., German, B. J. 2017. A convolutional neural network approach to training predictors for airfoil performance. 18th American Institute of Aeronautics nnd Astronautics/ISSMO Multidisciplinary Analysis and Optimization Conference, 3660.

[6] Zhang, Y., Sung, W. J., Mavris, D. 2018. Application of convolutional neural network to predict airfoil lift coefficient. American Institute of Aeronautics and Astronautics/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference.

[7] Zhu, L., Zhang, W., Kou, J., Liu, Y. 2019. Machine learning methods for turbulence modeling in subsonic flows around airfoils. Physics of Fluids, 31, 015105.

[8] Rehan, M., Javed, A., 2020. Prediction of Aerodynamic Coefficients and

Analysis of Flow Past a Supersonic Missile. Journal of Aeronautics and Aerospace Engineering, 9, 1-7.

[9] Viquerat, J., Hachem, E. 2019. A supervised neural network for drag prediction of arbitrary 2D shapes in low Reynolds number flows. arXiv preprint arXiv:1907.05090.

[10] Kutz, J. N. 2017. Deep learning in fluid dynamics. Journal of Fluid Mechanics, 814, 1–4.

[11] Santos, M., Mattos, B., Girardi, R., 2008. Aerodynamic coefficient prediction of airfoils using neural networks. In *46th AIAA aerospace sciences meeting and exhibit* (p. 887).

[12] Miyanawala, T. P., Jaimana, R. K. 2017. An efficient deep learning technique for the Navier-Stokes equations: Application to unsteady wake flow dynamics. arXiv preprint arXiv:1710.09099.

[13] Sun, G., Wang, S. 2019. A review of the artificial neural network surrogate modeling in aerodynamic design. Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering, 233, 5863–5872.

[14] Morimoto, M., Fukami, K., Zhang, K., Nair, A. G., Fukagata, K., 2021. Convolutional neural networks for fluid flow analysis: toward effective metamodeling and low-dimensionalization. Theoretical and Computational Fluid Dynamics, 35, 633-658.

[15] Li, J., Zhang, M., Martins, J. R. R. A., Shu, C. 2020. Efficient aerodynamic shape optimization with deep-learning-based geometric filtering. American Institute Of Aeronautics And Astronautics Journal, 58, 4243–4259.

[16] Tompson, J., Schlachter, K., Sprechmann, P., Perlin, K. 2017. Accelerating eulerian fluid simulation with convolutional networks. 34th International Conference on Machine Learning, ICML 2017, 7, 5258–5267.

[17] Wang, Y., Shimada, K., Farimani, A. B. 2021. Airfoil GAN: Encoding and Synthesizing Airfoils foraerodynamic-aware Shape Optimization. arXiv preprint arXiv:2101.04757

[18] Haizhou, W. U., Xuejun, L. I. U., Wei, A. N., Hongqiang, L. Y. U. (2022). A generative deep learning framework for airfoil flow field prediction with sparse data. Chinese Journal of Aeronautics, 35, 470-484.

[19] Duru, C., Alemdar, H., Baran, Ö. U. 2020. CNNFOIL: convolutional encoder-decoder modeling for pressure fields around airfoils. Neural Computing and Applications, 33, 6835-6849. Doi:https://doi.org/10.1007/s00521-020-05461-x.

[20] Thirumalainambi, R., Bardina, J. 2003. Training data requirement for a neural network to predict aerodynamic coefficients. Independent Component Analyses, Wavelets, and Neural Networks, 5102, 92-103.

[21] Yuan, Z., Wang, Y., Qiu, Y., Bai, J., Chen, G. 2018. Aerodynamic coefficient prediction of airfoils with convolutional neural network. In Asia-Pacific International Symposium on Aerospace Technology (pp. 34-46). Springer, Singapore.

[22] Karali, H., Demirezen, M. U., Yukselen, M. A., Inalhan, G. 2020. Design of a deep learning based nonlinear aerodynamic surrogate model for uavs. American Institute of Aeronautics And Astronautics Scitech 2020 Forum, 1 partf, 1–14.

[23] Fukami, K., Fukagata, K., Taira, K. 2020. Assessment of supervised machine learning methods for fluid flows. Theoretical and Computational Fluid Dynamics, 34, 497–519.

[24] Hui, X., Bai, J., Wang, H., Zhang, Y. (2020). Fast pressure distribution prediction of airfoils using deep learning. Aerospace Science and Technology, 105, 105949.

[25] Kim, J., Lee, C. 2020. Prediction of turbulent heat transfer using convolutional neural networks. Journal of Fluid Mechanics, 882, 1-37.

[26] Sun, D., Wang, Z., Qu, F., Bai, J. 2021. A deep learning based prediction approach for the supercritical airfoil at transonic speeds. Physics of Fluids, 33, 1-10.

[27] Yamashita, R., Nishio, M., Do, R. K. G., & Togashi, K. (2018). Convolutional neural networks: an overview and application in radiology. *Insights into imaging*, 9, 611-629.

[28] Marius-Constantin, P., Balas, V. E., Perescu-Popescu, L., Mastorakis, N. 2009. Multilayer perceptron and neural networks. WSEAS Transactions on Circuits and Systems, 8, 579–588.

[29] H, D. 2019. Mathematical Representation of a Perceptron Layer (with example in tensorflow). https://medium.com/@daniel.hellwig.p/mathematical-representation-of-a-perceptron-layer-with-example-in-tensorflow-754a38833b44

[30] Dinesh. 2019. CNN vs MLP for Image Classification. Medium. https://medium.com/analytics-vidhya/cnn-convolutional-neural-network-8d0a292b4498

[31] Lecun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., Jackel, L. D. (1989). Backpropagation Applied to Handwritten Zip Code Recognition. Neural Computation, 1, 541–551.

[32] Ke, Q., Liu, J., Bennamoun, M., An, S., Sohel, F., Boussaid, F. 2018. Computer vision for human-machine interaction. In Computer Vision For Assistive Healthcare. Elsevier Ltd.

[33] Colaboratory, G. 2021. Deep Learning Workshop. https://colab.research.google.com/github/rojiark/deeplearningworkshop/blob/master/deeplearningworkshop.ipynb

[34] IBM. 2020b. Pooling Layer. https://www.ibm.com/cloud/learn/convolutional-neural-networks

[35] IBM. 2020a. Fully-Connected Layer. https://www.ibm.com/cloud/learn/convolutional-neural-networks

[36] Chollet, F. 2018. Keras: The Python Deep Learning library. Astrophysics Source Code Library.

[37] Kingma, D. P., Ba, J. 2014. Adam: A Method for Stochastic Optimization. arXiv preprint arXiv:1412.6980.